Adobe® Flash® Media Server 3

The next generation of Adobe's award winning software for streaming video and real-time communication

Table of contents

1 Introduction
2 What is Flash Media Server 3?
3 The new Adobe Flash Media Server 3 family
5 Flash Media Server quick comparison
6 Flash Media Server system requirements
7 Flash Media Server upgrades
7 Flash platform for video delivery
8 How Flash Media Server 3 works
10 Extending Flash Media Server
11 Benefits of streaming versus HTTP delivery
12 Streaming delivery
15 Feature summary of Flash Media Server 3
21 Security features
23 HD (video quality) features
25 Live video features
26 Mobile delivery features
27 Programming features
28 Deploying Flash Media Server 3
29 Verifying installation
32 Configuring performance features
38 Configuring security features
40 Configuring general settings
42 Using server tools
42 Scaling Flash Media Server 3
46 Using live video
49 Securing content with Flash Media Server 3
50 Locking down your content
53 Glossary
54 Online resources
55 Flash Media Server community
55 References
55 About the author

The Adobe Flash Media family of products has become the industry-leading solution for streaming video and real-time communication. The ubiquity of the Adobe Flash platform provides a rich and seamless viewing experience across all operating systems. With the release of Flash Media Server 3 software, customers will benefit further from significant performance and scalability improvements.

This white paper will familiarize you with the powerful features of Flash Media Server 3, with special attention to the functionality and performance improvements in this version—including the new licensing structure and configuration, which makes assessing requirements and deploying Flash Media Server easier and more affordable. You will learn about the software's scalability and the benefits of streaming, and gain the knowledge you need to make informed choices about how to deliver and monetize video and communication services to the largest online audience.

Introduction

The Adobe Flash Media Server family of products provides the rich media delivery platform of choice that reaches more people, more securely and efficiently, than any other technology. From user-generated content to movies and television shows to corporate training, Adobe Flash Media Server offers enterprise-level solutions to deliver content and communications. Benefits include:

Superior user experience

- Faster video playback
- Live video support

Quality

- Streaming support for On2 VP6 and H.264/AAC video codecs
- Automatic bandwidth detection and dynamic buffering

Ubiquity of delivery

- Cross-platform client support (Windows,® Mac, and Linux®)
- Adobe Flash Player available regardless of platform—with over 8 million downloads every day and penetration on 99% of personal computers worldwide
- Windows or Linux server distributions

Interactivity

- Integrated video, audio, and data streaming for a complete social media toolset
- Customizable server (using the server-side ActionScript™ language and new plug-in architecture)

Adobe®

Security

- More secure content delivery with encryption and access control features and no client cache
- SWF verification to ensure applications are authorized

Support for your business

- Better use of bandwidth
- Optimized deployment profiles to significantly reduce deployment costs
- Large and active Flash/Flex® developer community
- Ability to measure streaming delivery with customized logging
- Playback experience controlled by the content owner/distributor
- Customer's brand a priority

These benefits and more make Flash Media Server 3 your best choice for delivery of your branded content—whether it's live or video on demand. Adobe has created a server that is robust, efficient, and competitively priced. And with its expanded scalability options, it can easily grow as your business grows.

The ubiquity of the Adobe Flash platform across all screens—from desktop computers to other devices—is a powerful advantage. Flash Media Server allows you to stream video to web browsers via Flash Player, to the desktop on the Adobe AIR™ platform, or to mobile and other devices through Flash Lite™ 3 software. This ubiquity allows you to fully monetize your video, reaching the most people with the least hassle.

Unlike other many other video delivery technologies, which just present pre-branded players to your viewers, Flash Media Server 3 integrates with Adobe Flash Player and Adobe AIR framework which allows you to create completely customized interfaces. Real-time data sharing, server-side plug-ins, logging, and monitoring application programming interfaces (APIs) provide developers and IT teams with the tools they need to develop and administer large-scale rich media applications.

Adobe Flash Media Server 3 provides:

- High-quality video experiences for consumers
- A ubiquitous and secure platform for publishers
- A rich multiway application for advertisers
- A scalable, enterprise streaming solution for IT professionals
- APIs to produce the next generation of multiway social media applications for developers

What is Flash Media Server 3?

Adobe Flash Media Server 3 is a scalable, real-time media server that delivers high quality (up to HD level), on-demand, live audio and video content with great efficiency and superior quality of service to reach the largest possible audience, regardless of the platform. It can deliver prerecorded video, live video, playlists, music, video blogging, video messaging, multimedia chat environments, real-time datacasting, multiuser gaming, and more.

Flash Media Server communicates and streams to Flash Player, Adobe AIR, Flash Lite 3, and the new Adobe Media Player consistently across platforms and browsers.

Flash Media Server 3 has many improvements and new features, including:

- New real-time encrypted communication channel RTMPE, an enhanced version of Adobe's Real Time Messaging Protocol (RTMP)
- New enhancements in content caching
- New connection throttling
- New authorization plug-in API for stream security
- Better quality of service for live streaming with data keyframes
- Better digital rights management (DRM) framework of encryption and access control
- New file adaptor API, allowing for remote content caching
- New support for mobile streaming to Flash Lite 3
- New Action Message Format (AMF) 3 support to send complex data to clients
- New Internet Protocol version 6 (IPv6) support
- Complete support for legacy Flash Media Server 2 applications
- Easier to deploy, with new services, improved documentation, and preconfiguration

These improvements represent a significant step forward, giving developers much-needed tools for creating powerful rich media applications. In addition, the expanded support documentation, easier publishing points, and sample applications that ship with Flash Media Server 3 flatten the learning curve and speed production.

The new Adobe Flash Media Server 3 family

There are three unique versions of Flash Media Server:

- Adobe Flash Media Development Server 3 (free from Adobe)
- Adobe Flash Media Streaming Server 3 (new version)
- Adobe Flash Media Interactive Server 3

The Macromedia® Flash Media Server 2 Professional, Origin, and Edge editions are now incorporated into Flash Media Interactive Server 3 as a native feature. This is a significant change, allowing Flash Media Server 3 to operate as either an origin or an edge server to distribute the traffic load. For more details on Origin/Edge configurations, see the "Scaling Flash Media Server 3" section.

Let's explore the features of each server to help you select the best solution for your specific application.

Adobe Flash Media Development Server 3

This free edition is available from *www.adobe.com*. It can be used in production for anyone who wants to implement basic low-volume streaming or social communication solutions. It can also be used for developing advanced streaming or social applications, since there is no functionality limit. It can even be used to leverage the new multipoint publish feature which allows you to create a live publishing point on your network, inject data messages into the stream, and then push the video to a larger content delivery network. There is a capacity limit on this free server of ten simultaneous inbound connections. The number of outbound connections is limited only by your server bandwidth and processing power.

Adobe Flash Media Streaming Server 3

Flash Media Streaming Server 3 is an economical solution that allows you to quickly start streaming high-quality, more secure video. It provides all the features you need to stream video and audio, and works in unison with Adobe Media Player and Adobe Flash Media Encoder. Core features include:

- Low-cost streaming to Flash, Adobe AIR, Adobe Media Player, and Flash Lite
- Easy to install and get started
- HD quality (with industry-standard H.264 video capabilities)
- Advanced tracking and reporting
- High performance—saturates a 1Gbps interface
- Network efficient
- Enhanced seeking
- Encrypted streaming
- Simple access control
- High-quality live video
- Multiple bitrate

Flash Media Streaming Server 3 ships with two services that make it easy to start streaming right out of the box:

- **Live video streaming**—The standard live video streaming application provides a publish point on Flash Media Streaming Server, allowing you to start publishing right away. It supports the Adobe Flash Media Encoder, the FLVPlayback component (Flash 8 and later), and ships with a sample live stream subscriber SWF.
- **Video on demand (VOD)**—The standard VOD application features server-to-client bandwidth detection, domain-based authentication, full support for the FLVPlayback component (Flash 8 and later), and ships with sample FLV, SWF, and HTML files for playback.

The server-side code for the Flash Media Services is fixed and cannot be modified. You can use the provided example media files for testing or upload your own, and run multiple instances.

Both applications support the new stream data access feature in Flash Media Server 3, which allows you to access the bitmap data of a stream, and will also work with the Flash Media Interactive Server edition. In the Flash Media Interactive Server edition, however, you can enhance the applications with custom functionality, record streams, utilize remote shared objects, and access additional scalability features. Refer to the Adobe Flash Media Services documentation for more information.

This server edition is ideal for one-way secure video streaming.

Adobe Flash Media Interactive Server 3

Adobe Flash Media Interactive Server 3 is the step-up from Flash Media Streaming Server that adds on advanced streaming features such as:

- Edge server caching
- Access control APIs
- Redirection support (similar to HTTP 302 redirection)
- Plug-ins

- Custom video services
- Server-side video recording
- Multiway communication
- Social media solutions
- Distributed core processing
- Multipoint publishing
- Live-stream metadata injection

Flash Media Interactive Server also lets you include value-added multiway solutions to help you socialize your streaming media with advanced real-time communication and collaboration services. Flash Media Interactive Server is the only high-performance and scalable server on the market that supports multiway applications including webcam video chat, recording, Voice over Internet Protocol (VoIP), and online games. Flash Media Interactive Server is the workhorse of the Flash Media family. It has all the features of the Flash Media Streaming Server, and more.

This server is ideal for:

- Large-scale deployment
- Developing custom video solutions
- Developing communication experiences
- Supplementing live or on-demand video streaming services with interactive features

Flash Media Interactive Server can also be used to interact with specialty data servers such as Lightweight Directory Access Protocol (LDAP) for authentication, Flash Remoting, Simple Object Access Protocol (SOAP), or XML, and integrates with Adobe Media Player and Adobe Flash Media Encoder. For even more customization, you can also implement custom server-side ActionScript, and develop plug-ins in C++ that further extend the functionality of the server.

Flash Media Server quick comparison

Features	Flash Media Interactive Server 3	Flash Media Streaming Server 3	Flash Media Development Server 3
• HD video support (H.264/AAC) • Real-time encrypted streaming • Robust logging • SWF verification • Live video streaming • Recorded video streaming • Network efficiencies • Maximized hardware capacity • Enhanced cache • 2GB file support • Built-in bandwidth detection • IPv6 support • Adobe Media Player support • Flash Lite 3 mobile support • Data keyframes • Stream data access	•	•	•
Simultaneous connections	Unlimited	Unlimited	10

Features	Flash Media Interactive Server 3	Flash Media Streaming Server 3	Flash Media Development Server 3
Bandwidth limitations	Unlimited	Unlimited	Unlimited
Processor limit	8-way SMP (cores)	4-way SMP (cores)	4-way SMP (cores)
Flash Media Server services (live and VOD)	●	● (Not customizable)	●
Process scopes and distributed cores	●		
Archive (record) video on server	●		●
Custom server-side applications (server-side ActionScript)	●		●
Edge server configuration	●		●
C++ plug-in support	●		●
Server-side playlists (Internet TV)	●		●
Multipoint publish/redirect	●		●
Remote shared objects	●		●
AMF3 support	●		●
Server redirection	●		●
Plug-in architecture for LDAP authentication	●		●

Flash Media Server system requirements

Windows	Linux
Microsoft® Windows Server® 2003 with Service Pack 1 (all 32-bit editions) Microsoft Windows XP (Flash Media Development Server only)	Linux Red Hat® 4 (32-bit only)
Hardware requirements	
• 3.2GHz Intel® Pentium® 4 processor (dual Intel Xeon® or faster recommended) • 2GB of RAM (4GB RAM recommended) • 1GB Ethernet card	

For the most up-to-date requirements, see *www.adobe.com/go/learn_fms_sysreqs_en*.

Flash Media Server upgrades

Upgrading from Macromedia Flash Media Server 2

The guidelines for upgrading from Flash Media Server 2 vary depending on your current license and when it was purchased:

- If you own any version of Flash Media Server 2 including Professional, Origin, or Edge, you can purchase an upgrade to Flash Media Interactive Server 3.
- Education licenses are not upgradable.
- If you purchased Flash Media Server 2 between December 4 through January 25, 2007, you will be eligible for a free upgrade to Flash Media Interactive Server 3. You must contact Adobe by March 31, 2008 to receive your license.
- Owners of Flash Communication Server 1.x (Personal and Professional Editions) are not eligible for upgrade licenses for version 3.0.
- There are no upgrade licenses available for Flash Media Streaming Server 3.

Note: To upgrade, you will need your Flash Media Server 2 serial number.

Upgrading from Flash Media Streaming Server

Upgrading from Flash Media Streaming Server to Flash Media Interactive Server only requires the purchase of an upgrade serial number; the server software itself will not need to be altered.

Note: Installing the upgrade will not include the additional Flash Media Interactive Server documentation; you will need to acquire the documentation from Adobe with your upgrade purchase.

For a comparison of features in each edition, see the Flash Media Server quick comparison table on the previous page.

Flash Media Server 3 Upgrade Plans

Flash Media Server 3 now has upgrade plans available that protect your investment for up to 2 years. Upgrade plans can help to ensure that you have the most recent version of Flash Media Server. Contact your Adobe Representative for pricing details and more information.

Flash platform for video delivery

Flash Media Server 3 can stream video and enable communications to more technology than ever before—with its ubiquitous in-browser platform using Flash Player, it can be deployed to the desktop with Adobe AIR and Adobe Media Player, and to mobile and other devices with Flash Lite 3.

Flash Media Server support in Adobe Flash Player (versions 6, 7, 8, 9 or later)

Features	Version 6	Version 7	Version 8	Version 9	Version 9,0,115,0
Sorenson Video (H.263): play and capture	•	•	•	•	•
On2 VP6 Video: play only			•	•	•
H.264 and High Efficiency-AAC: play only					•
RTMPE/RTMPTE					•
SWF verification					•

Adobe AIR (version 1.0 or later)

Adobe AIR is a cross-operating system runtime that enables you to use your existing HTML, Ajax, Flex, or Flash web development skills and tools to build and deploy rich Internet applications to the desktop.

Adobe AIR applications support native desktop integration, including clipboard and drag-and-drop support, local file input/output, system notification, and more. Adobe AIR applications can connect to Flash Media Server to stream audio and video, or share data, just as SWF files do.

Adobe Media Player (version 1.0 or later)

Adobe Media Player is a free stand-alone application built on the Adobe AIR platform that provides customized video delivery, branded channels, advanced usage tracking, and digital rights management support. Flash Media Interactive Server provides the mechanism that allows Adobe Media Player to stream media, track and log client events, and viewing history.

Adobe Flash Lite (version 3.0 or later)

Flash Media Server 3 can stream to devices that support the Flash Lite 3 mobile platform. This support opens up possibilities for interactive streaming to new markets, with powerful features:

- True FLV and MP3 streaming (On2 VP6-E, Sorenson Spark, MP3 codecs)
- ActionScript 2.0 API (NetConnection and NetStream)
- Device ID detection to allow optimization for specific devices
- Support for Real-Time Messaging Protocol (RTMP) tunneling

Flash Media Server 3 will support both prerecorded and live streaming. The same video experience available in the browser can now be delivered to mobile devices supporting Flash Lite 3.

How Flash Media Server 3 works

Flash Media Server solutions have both a server-side and a client-side architecture. The client experience is deployed as a SWF or AIR, created in either Flash or Flex. Clients run within a web browser (Flash Player), mobile device (Flash Lite 3) or as a desktop application (Adobe AIR). A client could also be another Flash Media Server, Adobe ColdFusion® 8, Adobe Flash Media Encoder, or licensed third-party technology that can stream or communicate with Flash Media Server. The server manages client connections and security, reads and writes to the server's file system, and performs other tasks.

The client is the initiator of the connection to the server. Once connected, the client can communicate with the server and with other connected clients. Clients connect to *instances* of applications; for example, a chat application may have many rooms. Each room is an instance of the chat application. Multiple instances of an application can be running simultaneously. Each application instance has its own unique name and provides unique resources to its connected clients.

Flash Media Server communication protocol (RTMP)

Flash Media Server communicates with its clients using the Adobe patented, RTMP over Transmission Control Protocol (TCP) that manages a two-way connection, allowing the server to send and receive video, audio, and data between client and server. In Flash Media Server 3, you also have the option to use stronger stream security with encrypted RTMP (RTMPE). RTMPE is easy to deploy and faster than using Secure Socket Layer (SSL) for stream encryption. RTMPE is just one of the robust new security features in Flash Media Server 3.

There are five configurations of RTMP with Flash Media Server 3:

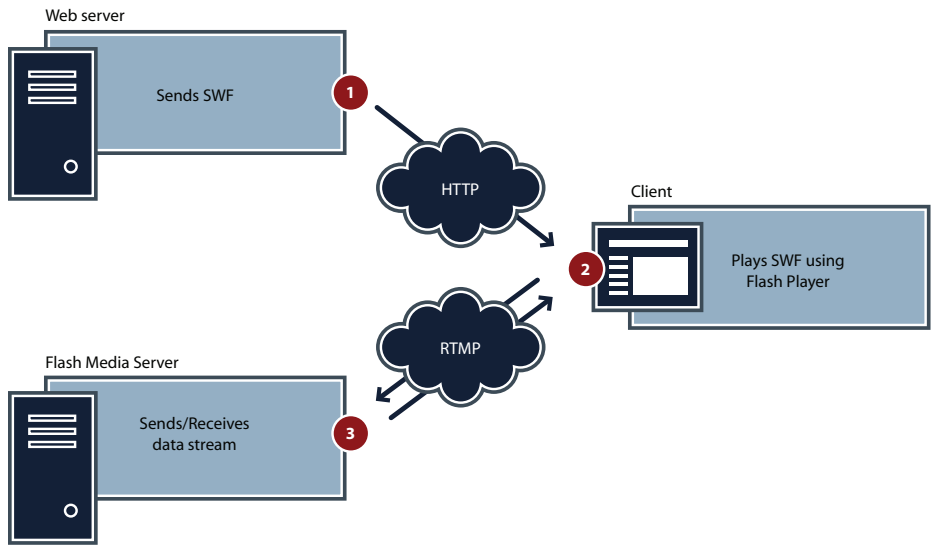
- **RTMP—**This is the standard, unencrypted RTMP. The default port is 1935; if a port is not specified, the client will attempt to connect to ports in the following order: 1935, 443, and then via RTMPT on port 80. Port 1935 is a registered IANA port.
- **RTMPT—**This protocol is RTMP "tunneled" over HTTP; this means that the RTMP data is encapsulated as valid HTTP data. The default port is 80.

- **RTMPS—**This protocol is RTMP sent over an SSL. SSL is a protocol that enables secure TCP/IP connections. (Flash Media Server natively supports both incoming and outgoing SSL connections.) The default port is 443.
- **RTMPE**—A new feature, this protocol is an enhanced and encrypted version of RTMP. RTMPE is faster than SSL, and does not require certificate management as SSL does. If you specify RTMPE without explicitly specifying a port, the Flash Player scans ports just as it does with standard RTMP, in the following order: 1935, 443, 80, and 80 (RTMPTE) (supported with Flash Player 9,0,115,0 or later; Adobe AIR; and Adobe Media Player).
- **RTMPTE—**A new feature in Flash Media Server 3, this enhanced version of RTMP encrypts the communication channel, tunneling over HTTP. The default port is 80 (supported with Flash Player 9,0,115,0 or later; Adobe AIR; and Adobe Media Player). The key benefits over SSL (RTMPS) are performance, ease of implementation, and limited impact on server capacity.

Utilizing the appropriate RTMP type, Flash Media Server can send streams through all but the most restrictive firewalls, and help protect rights-managed or sensitive content from piracy.

The following figure illustrates the basic architecture of a Flash Media Server connection.

Flash Media Server client/server architecture



Supported file types

Flash Media Server 3 is completely backwards-compatible with Flash Player 6 or above, Adobe AIR, and Flash Lite 3 clients. Additional formats and features are supported with newer versions of Flash Player.

Flash Media Server 3 continues support for FLV and MP3 media and AMF0 for data messaging. Flash Media Server 3 combined with Flash Player 9,0,115,0 now expands support for an industry-standard digital video format, MPEG-4.

The file formats supported by Flash Media Server are listed in the following table. All formats are supported by Adobe AIR.

File format	Type	Container	Flash Player version	Usual pairing
Sorenson Spark	Video	FLV	6, 7, 8, 9 or later	Nellymoser/MP3
On2 VP6	Video	FLV	Flash Lite 3: 8, 9 or later	Nellymoser / MP4
H.264*	Video	MPEG-4: mp4; m4v; f4v†; 3GPP	9,0,115,0 or later	AAC+/MP3
Nellymoser	Audio	FLV	6 or later	Sorenson Spark/ On2 VP6
MP3	Audio	MP3	Flash Lite 3: 6 or later	Sorenson Spark/ On2 VP6
AAC+, HE AAC, AAC v1, or AAC v2	Audio	MPEG-4: mp4; m4a; f4v; 3GPP	9,0,115,0 or later	H.264
AMF0	Data		Flash Lite 3: 6, 7, 8, 9 or later	
AMF3	Data		8, 9 or later	

*H.264 playback in Flash Player supports most popular profiles including Base, Main, and High Profile (HiP).

†The f4v format is a new format that will be a subset of MPEG-4 ISO 14496-10 and AAC+ (ISO 14496-3).

Note: To use H.264/AAC in Flash without any ActionScript, you will need the updated FLVPlayback component; it is available as an update to Flash CS3 Professional software. This update will also be required to use enhanced RTMP (RTMPE). Without the FLVPlayback component, developers can use ActionScript 1, 2, or 3 to create experiences with H.264.

For more information on H.264/AAC support, see the Flash Player 9 Update FAQ at *http://labs.adobe.com/wiki/index.php/Flash_Player:9:Update:H.264*.

Extending Flash Media Server

There are a number of APIs available for developing custom applications and monitoring of Flash Media Server.

Client-side ActionScript API

You can use any version of Adobe Flash (MX 2004 or later) or Adobe Flex Builder™ software to write client-side scripts that communicate with Flash Media Server (such as streaming or capturing live audio and video, or sending calls to server-side functions). These scripts can be any version of ActionScript.

Server-side ActionScript API

Flash Media Interactive Server provides access to Server-Side ActionScript. Server-Side ActionScript code can be used to control login policies, republish content to other servers, allow and disallow user access to server resources, allow users to update and share information, and more. Server-Side ActionScript is similar, but not identical to ActionScript 1.0.

Plug-in API

Flash Media Interactive Server offers plug-ins written in C++ that allow you to extend the functionality of the server. Some plug-ins perform access security checks, allow geographical targeting of content, and execute network-based file operations.

Administration API

The Administration API gives you the tools you need to create Flash Player or Adobe AIR clients that can monitor and administer all editions of Flash Media Server.

Benefits of streaming versus HTTP delivery

There are three methods for delivering video over the Internet using Adobe Flash Player:

- Embedded video
- Progressive download
- Streaming

Embedded video is rarely used except in very specialized applications with low-quality, short video clips, so our discussion will focus on progressive download and streaming.

In both progressive and streaming delivery, the video content is external to the SWF file. To deploy video content to the web, the SWF file and the video file would be uploaded to a server.

Keeping the video external and separate offers a number of benefits over the embedded video method, including:

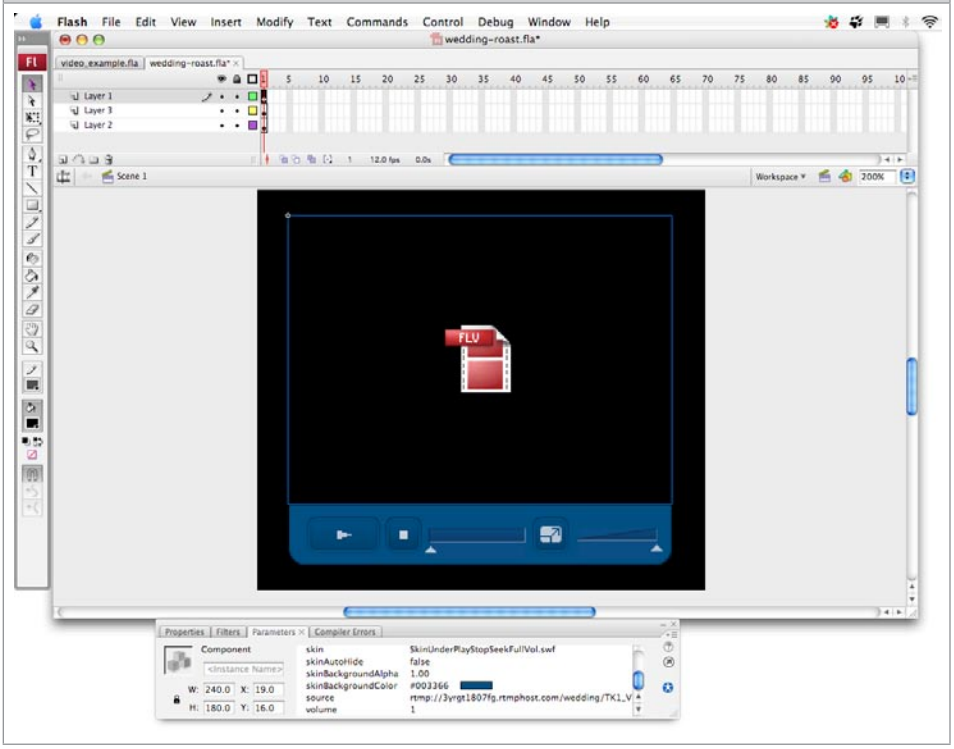
- Easy to update—Accommodates dynamic content and it's relatively easy to add or change content independent of the video player and without the need to republish the SWF file.
- Small SWF file size—Your SWF file can remain very small for faster page loading, allowing the video to be delivered when the user requests it.
- Better performance—Because the FLV and SWF files are separate, the user will have a better playback experience.

Note: Although this section focuses on the delivery of video files, the same methods can be used to deliver audio files. In other words, audio files can also be embedded, progressively downloaded, or streamed.

Progressive download video delivery

Since Flash MX2004, progressive download has been supported for video delivery. This method allows developers to load external video files into a Flash or Flex interface and play them back during runtime. This can be accomplished using ActionScript commands with the Video object or playback components, or by setting parameters for the playback components in the authoring environment (as shown in the following figure).

Parameters set for an FLVPlayback component to stream an external video file into a SWF



When the video is played, the video file first begins to download to the user's hard drive, then playback starts. The video will begin to play when enough of it has downloaded to the user's hard drive. The file is served from a standard web server through an HTTP request, just like a normal web page or any other downloadable document.

In comparison to streaming video, there's really only one consistent benefit to progressive download—you don't need a streaming server to deliver the video. Progressive download video can be served from any normal web server. While this can be convenient and potentially cost-effective, you should keep in mind that progressive downloads have limited seek and navigation capabilities and users can access and repurpose your content.

When to choose progressive download

Progressive download is a good choice for hobbyists or websites that have low traffic requirements, if they don't mind if their content is cached on a user's computer, and they only need to deliver shorter length videos (under 10 minutes). You will need to stream your video if you need advanced features and control over video delivery, and/or if you need to display video to larger audiences (for example, several hundred simultaneous viewers), track and report usage or viewing statistics, or if you want to offer your viewers the best interactive playback experience. Streaming delivery also consumes less bandwidth than progressive delivery, because only the portion of the video that is watched is actually delivered.

Streaming delivery

The ability to stream video and audio was first available with Flash MX, Flash Player 6, and Flash Communication Server MX.

When you use streaming delivery, the video files are kept external to the other content, just like progressive downloads. Developers can use ActionScript commands (and parameter settings with media components) to load external video files into a SWF file and play them back at runtime. In fact, the ActionScript code needed for streaming is almost identical to that for progressive download.

However, in streaming video, each client opens a persistent connection to the streaming server, and the server streams the video bits to the client. Those bits are displayed by the viewer and then immediately discarded.

This tight connection between the server and the client, and the server's ability to precisely control and deliver any portion of a stream as requested, enables the developer to take advantage of some advanced capabilities, including:

- Determining the client bandwidth and serving a stream with an appropriate bitrate.
- Measuring and tracking the stream's quality of delivery and switching to a lower (or higher) bitrate stream if needed (for example, if network congestion increases).
- Automatically generating thumbnails or playing short previews of your video clip without having to create separate images or video clips, and without downloading the entire video in the background.
- Automatically creating "chapters" (with appropriate thumbnails) that can be used for navigation of longer videos, without having to break the video into smaller files.
- Seemlessly switching midstream from one camera angle or one stream, to another.
- "Editing" video clips together to create one continuous video for playback. For example, you could play the first 10 seconds of clip 1, followed by the content between the 30- and 40-second marks of clip 2, followed by the last 20 seconds of clip 3.
- Creating webcast live events or recorded events where all viewers access the same content at the same time.

Why streaming is better

Progressive download is a simple method of video delivery with very little control—it's basically a simple HTTP download call. Streaming is a method that allows the publisher to control every aspect of the video experience.

The advantages of streaming video from Flash Media Server are numerous:

- **Fast start**—Streaming video is the fastest way to start playing any video on the web.
- **Advanced video control**—Features such as bandwidth detection, quality-of-service monitoring, automatic thumbnail creation, server-side playlists, and more.
- **Efficient use of network resources**—Customers who pay for their video hosting or bandwidth by the number of bits that are transferred can reduce their costs by using streaming video, since only the bits that the client actually views are transferred.
- **More secure, protected media delivery**—Because the media data is not saved to the client's cache when streamed, viewers can't retrieve the video or audio file from their temporary Internet files folder. There are also additional security features in Flash Media Server 3 that prevent stream ripping and other risks to your file's security. For more details, see the "Securing content with Flash Media Server 3" section.
- **Minimal use of client resources**—Resources such as memory and disk space are significantly reduced with streaming, because the clients do not need to download and store the entire file.
- **Tracking, reporting, and logging capabilities**—Because progressive download is a simple download of a file, you can't easily log specific relevant statistics such as how long the video was viewed, if the user navigated forward, backward, or paused the video, how many times the viewer played the video, if the viewer left the web page before the video completed playing, and so on. Streaming enables you to easily capture this important data.
- **Full seek and navigation**—Users can immediately seek to any point in the video and have it start playing immediately from that point. This makes streaming a great solution for longer playing videos or applications such as video blogging, classroom lectures, and conference sessions, where you may want to jump into the video at a specific point rather than requiring the viewer to watch it from the beginning.
- **Deep interactivity**—The precise control found in streaming enables developers to create extensive interaction in their video applications. For example, the ability to switch camera angles, have one video spawn another video, or the ability to seamlessly switch to alternate endings, are all enabled by streaming.
- **Live video**—Streaming provides the ability to deliver live video and audio from any connected webcam or DV camera, and even directly from some video cards, natively in Flash Player.
- **Video capture and record** (Flash Media Interactive Server only)—In addition to live streaming, Flash Media Server also gives you the ability to record video either in conjunction with the live stream (for example, archiving an event) or on its own (for example, video messaging).
- **Multiuser capabilities** (Flash Media Interactive Server only)—In addition to live one-to-many streaming, Flash Media Server also enables multiuser streaming of audio, video, and data for the creation of video communication applications.

While streaming may be perceived as being more difficult than progressive download, they're actually extremely similar—they both use the same components and the same ActionScript commands. Streaming just gives the developer more power to create rich, interactive video applications.

The only potential downside to streaming is that it requires special server software. Just as a robust data application would require you to install an application server in addition to your web server, robust media delivery applications require a streaming server in addition to the web server.

Customers with high-volume streaming needs, popular content, or critical uptime requirements who don't want to build their own infrastructure can get the benefits of streaming video in the Adobe Flash Player by utilizing a Flash Video Streaming Service. These Adobe partners offer load-balanced, redundant deployment of Flash Media Server over a reliable content delivery network. For more information about Flash Video Streaming Service (FVSS) partners, visit *www.adobe.com/go/fvss*.

When to choose streaming

You can use streaming with the Flash Media Server in situations where you need to:

- Deliver long files (greater than 30 seconds) or high-bitrate files (greater than 100Kbps)
- Perform bandwidth detection, allowing you to deliver the best quality video for the available hardware
- Quality-of-service monitoring
- Real-time tracking
- Provide real-time data sharing and interactivity to your video experiences
- Stream live video and/or audio
- Record video and/or audio
- Serve more streams with less bandwidth

If your website or blog relies heavily on video, audio, or real-time data sharing, you can give your user the best experience by using the features in Flash Media Server.

Delivery comparison

The following table provides a comparison of the three video delivery techniques in Flash Media Server.

	Embedded video	Progressive download	Streaming delivery
Encoding	Video and audio is encoded on import into Flash using a Sorenson Spark or VP6-E codec. Alternately, FLV files (encoded elsewhere) can be imported and placed on the Flash Timeline (re-encoding is not necessary).	Video files are encoded in either the built-in or stand-alone version of Adobe Flash Video Encoder, through Flash Video Exporter and a third-party nonlinear editing or encoding product, or using a standalone video encoding application such as Sorenson Squeeze or On2 Flix.	Same as progressive delivery. In addition, you can capture and record live video feeds from client-side webcams or DV cameras, or using Flash Media Encoder, and control live encoding variables such as bitrate, frames per second, and video playback size programmatically.
File size	SWF files contain both the video and audio streams as well as the Flash interface, resulting in a single, substantially larger file size.	SWF and video files are stored separately, resulting in a smaller SWF file size.	Same as progressive delivery.
Start time	Large SWF files often require users to wait before the video starts playing, resulting in a negative user experience.	Starts relatively quickly, after enough of the video has downloaded to begin playback.	Immediate. The fastest way to go from initial load to actually playing the video.
Timeline access	When embedded in the Flash Timeline, video appears on individual frames and can be treated like any other object on the Stage.	Video is played back only at runtime. Individual frames are not visible on the Stage. Timeline events can be triggered at selected times during video playback using ActionScript.	Same as progressive delivery.
Publishing	Each time a Flash movie is published or tested, the entire video file is republished. Changes to video files require manually re-importing the files into the Timeline.	Video files are only referenced at runtime. Publishing to SWF format is much faster than embedded video. Video files can be updated or modified without recompiling the SWF file.	Same as progressive video. You can dynamically pull video files from virtual locations, such as your storage area network (SAN) or a FVSS or other content delivery network (CDN).

	Embedded video	Progressive download	Streaming delivery
Frame rate	Video frame rate and SWF movie frame rate must be the same.	The video file can have a different frame rate than the SWF file.	Same as progressive delivery. Live video capture has programmable control over frame rate.
ActionScript access	Video playback and control is achieved by manipulating the movie's playback on the Timeline.	The NetStream class can be used to load, play, and pause external FLV files. Seek can also be performed on the portion of the video that has been downloaded.	Same as progressive delivery. Server-side ActionScript can also be used to provide additional functionality such as synchronization of streams, server-side playlists, smart delivery adjusted to client connection speed, and more.
Components	No video-specific components.	Media components (Flash 8 Professional and Flash CS3 Professional) can be used to set up and display external video and audio files together with transport controls (play, pause, seek, and so on.).	Same as progressive video. Also, you can use Flash Media Server communication components for streaming live and multiway video.
Seek and navigation ability	Requires the entire SWF file to be downloaded before user can seek or navigate the video.	User can only seek to portions of the video that have been downloaded.	User can seek anywhere at any time.
Web delivery	The entire SWF file must be downloaded to the client and loaded into memory to play back video.	Video files are progressively downloaded, cached, and then played from the local disk. The entire video clip need not fit in memory.	Video files are streamed from Flash Media Server, displayed on the client's screen, and then discarded from memory in a play-as-you-go method.
Performance	Audio and video synch is limited. Sync between audio and video will suffer after approximately 120 seconds of video. Total file duration is limited to available RAM on the playback system.	Improved performance over embedded SWF video, with higher resolution and more reliable audio synchronization. Provides best image quality, which is limited only by the amount of available hard drive space on the playback system.	Improved efficiency from a web delivery perspective, with optimal bitrate delivery on an as-needed basis to as many customers as necessary.
Control over video stream	None	None	Full control over what gets delivered to the clients and when. Advanced access control via server-side ActionScript.
Support for live video	No	No	Yes
Compatibility	Flash Player 6 or later	Flash Player 7 or later	Flash Player 6 or later

Feature summary of Flash Media Server 3

The Adobe Flash Media Server 3 family offers significant new features that make it easy to stream video with more performance, protection, and security than ever before. The ubiquity of the Flash Player, powerful server-side and client-side APIs, and more affordable pricing, make the Adobe Flash Media Server 3 family the obvious choice for both streaming video and robust interactive applications.

These are the key features and improvements to the Flash Media Server 3 family:

- **Performance**—Maximizes the capacity of your hardware and lowers deployment costs
- **Security**—Helps ensure the protection of your content
- **Quality**—Delivers the highest quality content, up to HD level
- **Live**—Provides the best service and an instant start for high-quality live video
- **Mobile delivery**—Creates one version of content for delivery to mobile and other devices

Flash Media Server 3 performance

Flash Media Server has significantly increased how many streams can be delivered concurrently with a single server license. Delivering more streams requires fewer servers and lowers the cost of deployment when compared with Flash Media Server 2. Linux deployments have been improved by over 300% over version 2.

There are numerous features that contribute to increased performance. The features discussed in this section include:

- General performance optimization
- Connection throttling
- Enhanced process scopes
- Auto-close idle connections
- Enhanced RTMP (RTMPE)
- Built-in bandwidth detection
- Process scopes
- Distributed cores

Factors affecting performance

This section explains how performance for Flash Media Server 3 is affected by different conditions. Measuring performance increases is done by comparing the number of concurrent streams for a given CPU utilization. Knowing how many streams a server will support will help you understand how many servers you need to deploy.

The number of streams a server can deliver is dependent upon a number of conditions, including:

- **Protocol**—RTMP is the highest performing protocol, followed by RTMPE.
- **Video bitrate (quality)**—The lower the bitrate of your video, the more concurrent streams can be delivered by your server.
- **Platform**—You can deliver more connections with less CPU usage utilizing Linux Red Hat 4.
- **Hardware**—Hardware such as RAM, disk speed, CPU, and network speed will all influence the streaming capacity of Flash Media Server 3.
- **Configuration**—Flash Media Server 3 comes preconfigured for optimal streaming performance for most situations. Changing the configuration may improve your performance.
- **Application complexity**—If you deploy custom plug-ins or develop complex server-side application logic, your performance may increase or decrease.
- **Usage**—The way your users interact with your video will impact the server performance. Interactions could include connecting, disconnecting, seeking, or pausing. This will be discussed in more detail in the next section.

Flash Media Server 3 performance metrics

Flash Media Server 3 has over two times the performance on Windows deployments and over three times the performance on Linux. The following tables show the performance metrics for Flash Media Server in two scenarios:

- **VOD**—Using prerecorded video
- **Live video**—Using live video streamed from Flash Media Encoder 2

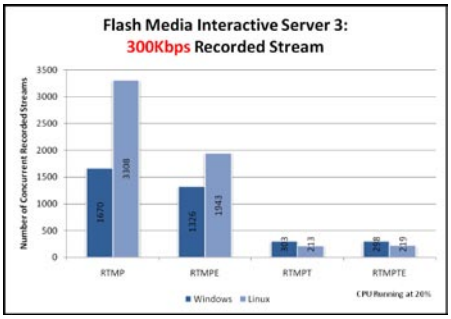
Capacity numbers were calculated using the following hardware platforms. The numbers in this study were achieved with 2 X 1 Gbps network adaptors, but the results published are limited to 1Gbps throughput.

Server hardware	HP ProLiant DL360 G4p servers, with dual-core 3.6GHz, Xeon processors with hyperthreading Server configuration: 3.5GB of 200MHz DDR SDRAM, a SCSI2, 64GB RAID 0 disk storage at 10K RPM, and bonded/teamed 1GB Intel Pro 100 XF Ethernet cards at 133MHz
Operating systems	Windows Server 2003 with Service Pack 1; Linux Red Hat 4, kernel 2.6.9-22
Flash Player	Version 9,0,115,0
Test media	• 700Kbps: FLV, 117 minutes, 763MB using On2 VP6 codec • 300Kbps: FLV, 53 minutes, 105MB using On2 VP6 codec • 128Kbps: MP3; 52 minutes, 61MB using MP3 ID3v2.3 tag
Flash Media Server	Version 3: default configuration using default chunk size and cache

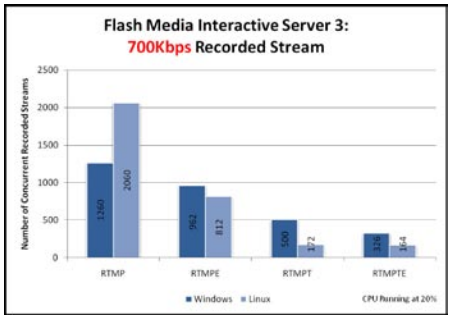
Video on demand

To provide a good comparison, the following graph shows the total number of streams achieved while using only 20% CPU. Linux was able to saturate a 1Gbps network adaptor with just over 20% CPU. The second graph illustrates the capability of each protocol with higher CPU usage.

300Kbps recorded video limited to 1Gbps

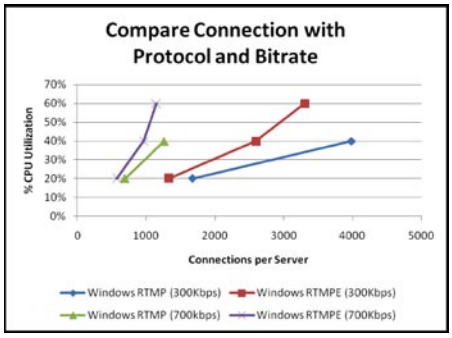


700Kbps recorded video limited to 1Gbps

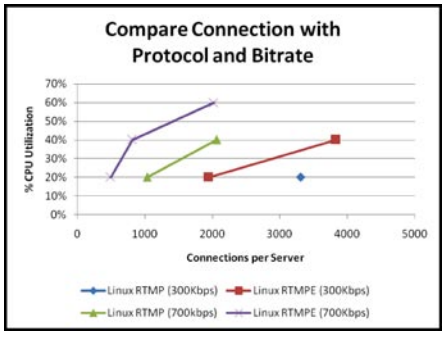


The following graphs show the number of concurrent streams given different percent CPU utilization, protocol, and bitrate. Notice that with more percent CPU utilization, you can deliver more streams. The graphs were limited to 1Gbps and never reached 100% CPU. Higher data rates resulted in faster saturation, while lower data rates used more CPU to deliver similar connections.

Windows



Linux

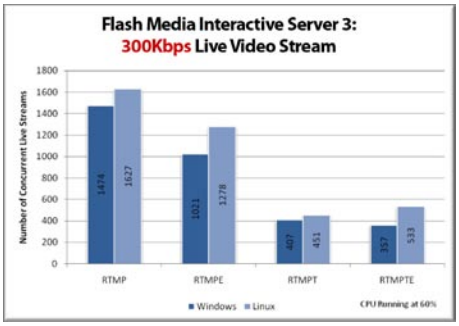


The impact of RTMPE reduced the capacity by only 25% to 30% on average, given similar percent CPU usage. If you are deploying RTMPE, you can expect increased CPU usage, but you will still be able to saturate a 1Gbps network with less then 70% CPU.

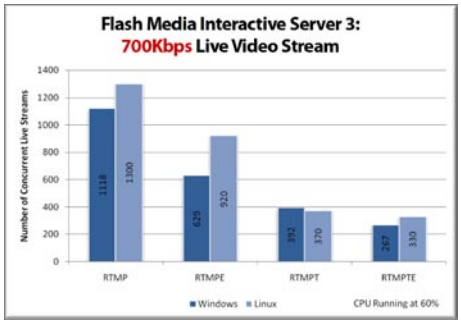
Live video

Live video streaming from Flash Media Server is impacted by the same properties as prerecorded video streamed on demand. The following graphs show the total capacity achieved with live video. The tests (see the following graphs) were done with live video streamed from Flash Media Encoder 2 using On2 VP6 video and MP3 audio codecs.

300Kbps live video



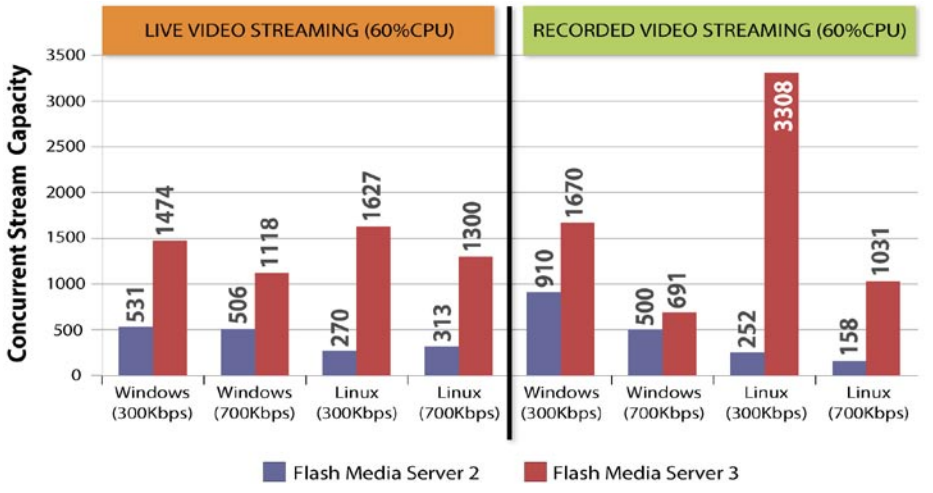
700Kbps live video



Comparing with Flash Media Server 2

Flash Media Server 3 nearly doubles the total number of active streams from Flash Media Server 2. Increases can be found in both live and prerecorded video applications and allow for significantly more connections at lower bitrates than Flash Media Server 2.

Total concurrent video stream capacity between Flash Media Server version 2 and version 3



All tests were sampled using RTMP. Values for live streaming were sampled at 60% CPU; values for recorded on-demand streaming were sampled at 20% CPU.

Connection throttling

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

Flash Media Server 3 features new connection-handling management that ensures high-quality service for users who are already connected to the server. In the past, if there were a large number of new users trying to connect to a popular application, the current users could experience a disruption of playback. Connection throttling provides a number of methods to preserve quality of service:

- Restricts the number of threads that handle incoming connections.
- Provides a configurable maximum rate (per second) at which the server will accept new connections. Connections that exceed this maximum rate are delayed, and queued in the operating system's network stack. The OS limits the number of connections it will keep in its queue. Once this queue is full, clients attempting to connect will then be rejected. (Note that this maximum rate is per "listener." For example, if your server is configured to listen on ports 1935 and 80, the connection rate would apply to each port. So if the connection rate was set to 10 connections per second, that could potentially mean a total maximum connection rate of 20 per second.)
- Provides a maximum queue length. Beyond this queue length, the server will reject connections to maintain quality of service for currently connected users.

These settings are, of course, fully configurable by the server administrator.

Enhanced process scopes

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

Flash Media Server 3 offers a greater degree of flexibility in configuring server process scopes. In Flash Media Server 2, a server administrator had three choices for process scope management: run a process for each virtual host, each application, or each instance. This could easily require a large number of processes. This new feature addresses this by limiting the number of processes to a predetermined number. This number of processes will then be distributed between all active virtual hosts by the server. For example, you could choose to have 10 processes, and if you had 20 active virtual hosts, they would be distributed automatically over the 10 processes. This results in a much more robust and scalable solution for process distribution.

Auto-close idle connections

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

In previous versions of Flash Media Server, the connection between the client and the server was controlled by the client. This meant, if the client disconnected unexpectedly, connections could be left open indefinitely. In Flash Media Server 3, the server can now detect and close these long-standing idle connections. This idle time can be set in the `Server.xml` file, with a default disconnect time of 60 seconds.

Enhanced RTMP

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 9,0,115,0 or later.

RTMP is the patented protocol used by Flash Media Server to send and receive data. In Flash Media Server 3, RTMP has been enhanced with performance improvements and increased security features.

RTMP in Flash Media Server 3 has been retooled to be more secure and more efficient. For security, the RTMP handshake between the Flash Player client and Flash Media Server 3 is more strictly enforced. SWF verification is now possible, ensuring that the SWF connection request is coming from the expected source. This helps to prevent FLV and bandwidth theft.

In previous versions of Flash Media Server, utilizing an SSL was the only option to encrypt your stream data. However, this resulted in a noticeably slower connection. The new RTMPE in Flash Media Server 3 secures the channel with 128-bit encryption between the client and the server without the performance degradation of SSL, and without the need for a certificate.

Similar to the implementation of SSL (RTMPS), you'll only need to specify RTMPE in your client's connection string to utilize the protocol. For example:

```
nc.connect("rtmpe://www.example.com/myApplication")
```

RTMPE behaves exactly like RTMP, but is encrypted. Additionally, you can request an encrypted tunneling connection by specifying RTMPTE.

Built-in bandwidth detection

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

Bandwidth detection is built into Flash Media Server, and is enabled by default. The new bandwidth detection, called native bandwidth detection, provides better performance and scalability than scripted bandwidth detection. To use native bandwidth detection, simply make sure bandwidth detection is enabled, and write client code that calls functions built into Flash Media Server. This feature is compatible with ActionScript 2 and 3, and no server-side code is required.

You can also choose to disable native bandwidth detection and implement detection in a server-side script, especially if you want to reuse existing code. To use server-side bandwidth detection, use the specialized `main.asc` file Adobe provides for bandwidth detection and disable native bandwidth detection in the `Application.xml` file. For more information, refer to the *Adobe Flash Media Interactive Server 3 Developer Guide*.

Process scopes

New feature in Flash Media Interactive Server; requires Flash Player 6 or later.

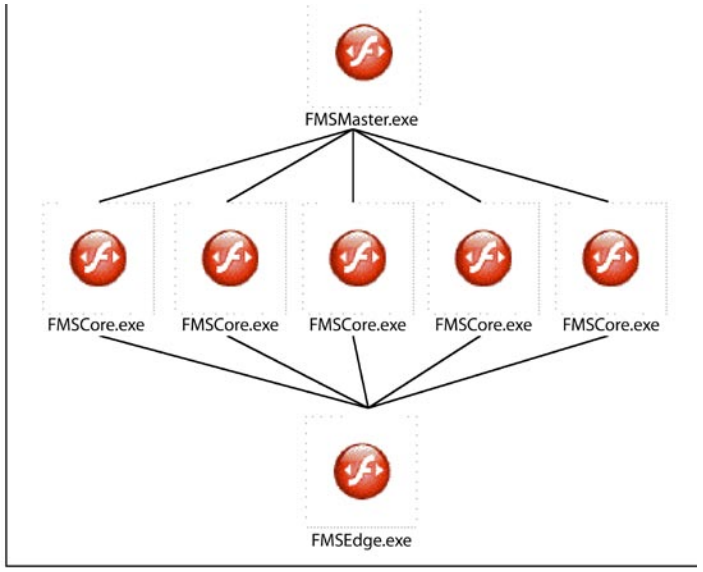
Flash Media Interactive Server 3 gives you control over how server processes are allocated. When you start the server, you are starting a process called `FMSMaster.exe` (Windows) or `fmsmaster` (Linux). Application instances run in processes called `FMSCore.exe` (Windows) or `fmscore` (Linux).

Flash Media Server operates with multiple processes. With the default installation, there will be four processes running: master, Edge, core, and admin.

The master process is a monitor that starts core processes when necessary. There can only be one master process running at a time, but there can be many core processes running at a time. You can configure how applications are assigned to server processes in the `Application.xml` file. You can specify the number of processes, the process scope, how long a process runs, and the number of process failures allowed before a core process is disabled.

Clients always connect through the Edge process (whether an Edge/Origin configuration exists or not). The master process is responsible for spawning and rolling over each core process. Clients never connect to the master process, and the master process cannot be configured. The following figure illustrates the master, core, and Edge processes.

Illustration of process scopes



Distributed cores

New feature in Flash Media Interactive Server; requires Flash Player 6 or later.

To further increase the capacity and reliability of your server, you can distribute connections across multiple processes for a specific scope. For example, if your scope was set to "adaptor," you could have connections spread across any number of core processes for each virtual host.

Like process scopes, the distributed core feature lets you increase the capacity of your server. Distributed cores will let you engage more RAM for caching and more threading for the process-intensive connection routine. They cannot be used when deploying a multiway hybrid or live solution because connections need to be on the same core process to share communication.

Security features

SWF verification

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 9,0,115,0 or later.

SWF verification is a new security feature in Flash Media Server 3 that allows you to directly control which SWF files can connect to your server. Without implementing this feature, any SWF with the proper connection Uniform Resource Identifier (URI) and application name could freely connect, potentially accessing your streams and using server resources.

With SWF verification, you can configure the server to check that the SWF file attempting to access a certain application or application instance belongs to a group of preapproved SWF files. Enabling this feature is easy. You simply store a copy of the approved SWF in the application directory and turn on the feature in the `Application.xml` file. When a SWF file connects to the server, the server verifies that the file exactly matches the SWF file in your application directory, and then accepts the connection.

To approve a SWF for any instance of a certain application, you'd place it in the `SWFs` directory in that application's folder; to approve a SWF for a specific instance, you'd place it in the `SWFs` directory inside that instance folder.

Note: If you're deploying an Adobe AIR application, copy the SWF file that you compiled into the Adobe AIR package, to the server to make it available for SWF verification.

Standardized server redirection handling

New feature in Flash Media Interactive Server; requires Flash Player 6 or later.

Flash Media Server 3 now supports stream redirection in RTMP, which behaves much like HTTP 302 redirection. This feature can be enabled by using an access adaptor server-side plug-in, or in server-side ActionScript. For example, while using an access adaptor running on an Edge server, you could use this redirect method to notify a client that a requested video does not exist in the expected location, and pass them a new URI for the stream. It could also be useful for other Flash Media Server load-balancing schemes, as well as content organization.

There is a revised server-side and client-side API that allows you to manage server redirection data. Note that you will need to use the updated FLVPlayback component to be able to take advantage of this feature. For more information, refer to the Flash Media Server 3 documentation, *Extending Flash Media Server.*

Server-side plug-in architecture

New feature in Flash Media Interactive Server; requires Flash Player 6 or later.

Adobe Flash Media Interactive Server 3 supports plug-ins written in C++ that you can customize to extend the server's functionality. There are three plug-in classifications: File, Authorization, and Access.

Each of these plug-ins can be used as-is, or customized to your specific needs; sample or skeleton API examples are provided that can be extended to meet your functional requirements. There can be more than one version of each plug-in. For detailed information, see the *Adobe Flash Media Server Plug-in API Reference.*

File plug-in

The File plug-in allows developers to write custom asynchronous functionality allowing complete control over where and how Flash Media Server reads content from any file system or service. The file system could be local or remote; it can be over any protocol, such as HTTP or even FTP.

Asynchronous read and write is a major new enhancement to Flash Media Server 3. Previous versions of Flash Media Server supported only synchronous access to a localized file system. Each request for a read operation on a file had to wait for the previous requests in the queue to be completed. The File plug-in builds on this new asynchronous access, making it easier to implement network-based and remote file I/O.

For example, you could retrieve files from a remote location over HTTP and serve them to clients via Flash Media Server. If you are about to read a file from the disk to stream, you can now read that file from any mapped location, rather than only from the streams folder of your current Flash Media Server application. This feature is only applicable to VOD content.

Authorization plug-in

The Authorization plug-in allows you to tightly control client access to server `NetConnection` and `NetStream` events. You can use this plug-in to perform tasks such as:

- Authorize connections to the server
- Authorize publishing, playback, or seeking within a stream
- Map logical stream requests to physical stream requests
- Apply rights management policies to stream requests
- Disconnect clients from the server
- Deliver content to clients according to their geographic location, subscription level, time, and duration of a specific user's access to specific streams, and so on

For example, an application with different membership levels could use the Authorization plug-in to deliver a high-definition stream to a paid member, and a standard definition stream to a guest. You would intercept the client before they connect, determine their membership level, then filter the connection to point to the correct stream file. This plug-in could also be used for access monitoring, logging, or implementing other custom rights management schemes. Since these Authorization plug-ins can be chained, you can implement a sequence of actions to create a sophisticated access filter for your content.

Access plug-in

The Access plug-in gives developers another security tool. It provides an additional layer between client and server, intercepting connection requests and examining both the client and the server to determine how to handle the request. With the Access plug-in, you can perform tasks such as:

- Query a database of login data to authorize the connection. If accepted, you could update the database with information about the client and keep a record of the connection.
- Determine the current server load, and choose to accept, reject, or redirect the client.
- Set read and write access for files and folders on the server.
- Set access to audio and video bitmap data on streams.

Enhanced cache

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

Caching behavior has been reconfigured, improving Flash Media Server memory management and server performance. The efficiency has been significantly improved, and the server administrator now has the power to set a ceiling on the amount of RAM utilized by the cache.

In Flash Media Server 2, the cache settings allowed you to specify the number of files you wanted to cache in the cache folder. Each of these files had a predetermined number of segments available to write into. If the server was too busy, all of the segments were not used in each file. Now, the cache is based only on segments, not the number of files—resulting in much more efficient caching behavior.

IPv6 compliance

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

Required by many government customers, IPv6 is the next generation protocol replacing IPv4 (such as, 192.168.0.1). This new version expands addressing capabilities from 32 bits to 128 bits, and is now supported in Flash Media Server 3. The Flash Media Server Administration Console will also be fully compatible with IPv6 addressing.

HD (video quality) features

Flash Media Server 3 has no limits on data rate or video quality. Flash Media Server 3 supports traditional FLV (using Sorenson Spark or On2 VP6/MP3 codecs) and now supports MPEG-4 (using H.264/AAC+ codecs). Whether you need to stream 50Kbps media files to dial-up connections and mobile devices or up to 20Mbps for full HD quality—Flash Media Server can support your application.

Video profiles in Flash

The video profile you select affects the quality of your video. Flash technology is organized around three video profiles: Light (LT), Standard (SD), and High Definition (HD).

- Video LT—Mobile delivery via Flash Lite 3 (On2 VP6 codec only)
- Video SD—Standard-definition web video (On2 VP6 and H.264 codecs)
- Video HD—High-definition video (On2 VP6 and H.264)

Each profile is further organized into three levels, as shown in the following table.

Level	Codec	Target resolution/data rate
LT level 1	VP6 Mobile	GSM
LT level 2	VP6 Mobile	3G
LT level 3	VP6 Mobile	3G-high
SD level 1	On2 VP6 and H.264	160x112
SD level 2	On2 VP6 and H.264	320x240
SD level 3	On2 VP6 and H.264	640x480
HD level 1	On2 VP6 and H.264	640x480
HD level 2	On2 VP6 and H.264	1,280x720
HD level 3	On2 VP6 and H.264	1,920x1,080

(Source: *www.adobe.com/products/hdvideo/supported_technologies/h264.html*)

Typical frame rates range from 5 fps to 30 fps. Higher frame rates and screen resolutions require more computing power to play back. Flash Player 9 supports hardware-accelerated full-screen video playback.

H.264 video codec

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 9,0,115,0 or later.

The H.264 codec delivers excellent quality video, and is now supported by Flash Media Server 3. The video streaming-related subsets of the MPEG-4 part 10 standard supported by Flash technologies are:

- Baseline—Widely used in videoconferencing and mobile applications running on devices with limited computing power.
- Main Profile (MP)—The original profile intended for broadcast and storage applications, MP has been largely overshadowed by High Profile.
- High Profile (HiP)—The primary profile for broadcast and disk storage applications. HiP is the profile adopted by both high-definition DVD formats: HD DVD and Blu-ray disc.
- High 10 Profile (Hi10P)—A profile that increases decoded picture precision of HiP to 10 bits per sample.

In addition to FLV, Flash Player 9 supports streaming or progressive playback of MPEG-4 container formats such as MP4, M4A, MOV, MP4V, 3GP, and 3G2 if they contain H.264 video and/or HE-AAC audio.

For more information about H.264 implementation, visit *www.adobe.com/products/hdvideo/supported_technologies/h264.html*.

For full system requirements for rendering HD content in Flash player, visit *www.adobe.com/products/hdvideo/systemreqs.html*.

HE-AAC audio codec

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 9,0,115,0 or later.

Flash Media Server 3 now supports streaming of HE-AAC audio. The codec profiles defined by the ISO/IEC 14496-3 (MPEG-4 part 3) standard are supported:

- **Advanced Audio Coding (AAC) Main—**Adds perceptual noise shaping to the MPEG-2 version of AAC, improving quality at lower bitrates. Can handle up to five channels plus one subwoofer channel (5.1) in a single audio object.

- **AAC Low Complexity (LC)—**Slightly less efficient than AAC Main and requires less CPU power to encode and decode. AAC LC is optimized for low-bitrate applications such as streaming.
- **High Efficiency AAC v2 (also known as HE-AAC+, eAAC, and aacPlus v2)—**A superset of the AAC core codec that combines spectral band replication (SBR) and parametric stereo (PS) techniques to enhance coding , especially for low-bitrate stereo signals. HE-AAC v2 supports up to 48 audio channels and enables 5.1 and 7.1 surround sound.

For more information about HE-AAC implementation, visit *www.adobe.com/products/hdvideo/supported_technologies/heaacv2.html*.

On2 VP6-S support

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 9,0,115,0 or later.

Flash Media Server 3 introduces streaming support for the new VP6-S codec profile available in Flash Player 9,0,115,0. VP6-S features greater simplicity in encoding/decoding of high-resolution, full-screen video, which allows high-definition video to be streamed and played back smoothly on mass-market computers with limited processor speeds. VP6-S is recommended for video delivered at or above 500Kbps at high resolutions when targeting these slower machines.

For more information on this codec, visit *www.on2.com*.

Smart buffering (player fix)

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 9,0,115,0 or later.

Flash Player 9,0,115,0 introduced an enhancement to the playback engine to sustain the buffer of a Flash Media Server stream when you pause playback of a recorded stream. This means that Flash Media Server will no longer flush the buffer when a video is paused. When resumed, playback will be immediate and not need to be rebuffered. This feature will allow you to prebuffer video and create seamless video switching among playlist items.

Live video features

Flash Media Server 3 introduces several powerful new features to enhance your live video publishing applications.

Live instant-on

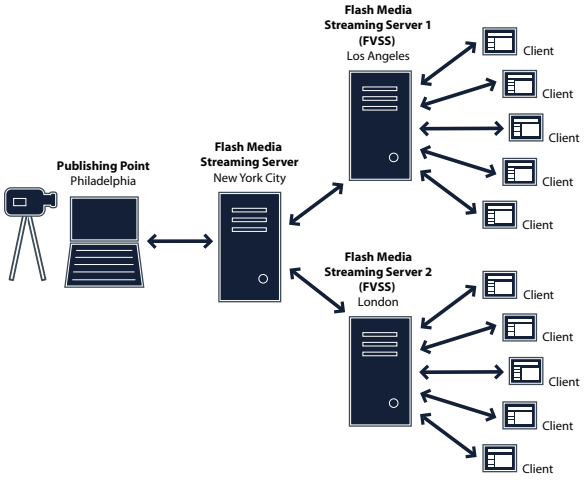
Video streaming through Flash Media Server starts playing instantly when a page is loaded or the play button is selected. This is a distinct benefit over other streaming technologies, as well as progressive delivery, and is compatible with both live and on-demand streams.

Multipoint publish

New feature in Flash Media Interactive Server; requires Flash Player 6 or later.

A new feature available in all editions of Flash Media Server 3, the multipoint publish feature gives flexibility and scalability to your streaming applications. Previously, if you were using a content delivery network (CDN) to deliver your streaming content, you were unable to implement any custom server-side code or inject any data messages into the outbound stream. Now, with multipoint publishing, you can use your own Flash Media Server (or Flash Media Encoder) to control the feed to the CDN, which then broadcasts it to your clients (as shown in the following figure). (The free development edition can actually be used in commercial applications as this local live publishing point.)

Multipoint publishing architecture



Stream data access

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 9,0,115,0 or later.

Stream data access allows you to control the ability to take a snapshot of a streaming video as a bitmap, on a per-client basis. For example, you can create dynamic thumbnails and video previews of streaming content.

Stream data access can be configured in your server-side code. Similar to the `readAccess` and `writeAccess` properties, you can now set `audioSampleAccess` and `videoSampleAccess` on streams. Flash Media Server will determine the permission setting for each stream and send a special data message to the player. (Since this feature uses bitmap data, it is only available with Flash Player 9 or later.)

Data keyframes for live video

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

One of the challenges in live video broadcast is the need for current stream metadata to be sent to viewers who are connecting midstream. Unlike an on-demand stream, where metadata can always be at the beginning of the stream and received when a user first subscribes, live streams can be subscribed to at any time. Therefore, these latecomers may never receive the live stream's metadata. Data keyframes eliminate this issue by sending metadata to new subscribers when they join the stream.

Mobile delivery features

Flash Media Server 3 can stream video in FLV (On VP6/MP3) format to mobile devices with Flash Lite 3 installed. Flash Lite 3 has limited implementation of the `NetConnection` and `NetStream` classes that only allow it to receive video, audio, and metadata packets.

When Flash Lite 3 connects with Flash Media Server, it will send user agent data. You can then use this information to filter access, or to deliver video that is optimized for the device. The user agent string sent by Flash Lite 3 consists of:

- Software family and version—For example, Flash Lite, version 3.0
- Device identification—For example, a Motorola RAZR V3x mobile phone
- Profile identification—Profile/specific Resource Description Framework (RDF) URI reference for the device
- Network/type—For example, network/2G, Code Division Multiple Access (CDMA), 3G, and so on

A user agent string might look something like this:

```
FlashLite/3.0.1 Device/RAZRV3x Profile/razrv3x Network/2.5G
```

Device information can be found in the RDF reference. A sample URL for the Motorola Razor is: *http://motorola.handango.com/phoneconfig/razrv3x/Profile/razrv3x.rdf.*

Flash Media Server can use this Flash Lite 3 information in numerous ways.

- Virtual keys—Virtual keys let you configure Flash Media Server (without any programming) to automatically access video from an alternative folder. Video in this folder could be optimized for the client. Traditionally this was used to filter clients that could not support the On2 VP6 codec. Now it can be used for Flash Lite.
- Authorization plug-in—The Authorization plug in can be developed to respond to the user agent by redirecting the request to an optimized video stream, or to an alternate stream if it's not available.
- Server-side ActionScript—Using the Client object on the server, you can access the user agent, parse it, and create simple access controls that manage what devices can access the video.

Programming features

Built-in services: live and VOD

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

There are two built-in applications, or services, that ship with both Flash Media Streaming Server and Flash Media Interactive Server:

- **VOD—**The VOD service is a publishing point that lets you upload video and audio files to your server and start streaming them without having to build a custom service or configure the server in any way.
- **Live—**The live service is a publishing point that lets you use Flash Media Encoder to easily stream live video without any custom server-side code or configuration.

Using one of these built-in services is the simplest way to start streaming using Flash Media Server. These services are available in all Flash Media Server versions. Detailed, step-by-step instructions for using these applications can be found in the *Flash Media Server Installation Guide*.

AMF3 support

New feature in Flash Media Interactive Server; requires Flash Player 8 or later.

Flash Media Server now supports the AMF3 format for sending data between the server and connected clients. All basic data types can now be serialized and deserialized by Flash Media Server, including Number, Boolean, String, Null, Undefined, Array, Date, XML, Object, and ByteArray.

Note: ByteArray data cannot be created or inspected using server-side scripting, but can be safely exchanged between clients.

This implementation is fully backward-compatible. Utilizing "blended mode" AMF support, both AMF3 and AMF0 clients will be allowed to connect to an application simultaneously.

If an AMF3 message is sent, however, the server will disconnect all clients that support only AMF0 data. For example:

- If clients are either all AMF0 or all AMF3, they are able to communicate with each other regardless of an application's encoding.
- If two clients with different encodings connect to a "myAMF" application, as long as the data being communicated does not contain any new ActionScript 3 datatypes (for example, ByteArray or XML) or externalized AMF3 objects, both clients will be able to communicate data to each other.

- If the AMF3 client decides to send a ByteArray, for example, then the AMF0 client will be disconnected, since AMF0 clients don't understand ByteArrays. (When an AMF0 client is disconnected because of incompatible encoding, this event will be logged in the server's log files.)

Wherever the server needs to serialize data for clients, it will generally attempt to serialize data first in AMF0. If that is not possible, the server will automatically serialize the data in AMF3. This support is available in all of the basic communication methods: `NetConnection.call`, `Client.call`, `application.broadcastMsg`, `SharedObjects`, `NetStream.send`, and data embedded in video files. It's also important to note that a server-to-server `NetConnection` (through server-side ActionScript) will default to AMF3.

In addition, default object encoding can be set in the `Application.xml` file. This can also be overridden for each individual `NetConnection` via the `NetConnection.objectEncoding` property. For example, the `Application.xml` may say AMF3, but you can set a `NetConnection` to be AMF0 by setting this property, or vice versa.

The addition of AMF3 support gives Flash Media Server 3 a much more flexible tool for data sharing, and opens up even richer possibilities for interactive applications—in many cases eliminating the need to integrate with other backend server technologies.

Administration API

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

The Administration API can be used to create custom tools to monitor, configure, and manage Flash Media Server 3. The Administration API methods can be called over HTTP via a web client, or via a Flash Player or Adobe AIR client over RTMP from any version of client-side ActionScript. Administration methods can be used to perform tasks such as adding or removing administration accounts, server configuration, garbage collection, virtual host administration, application and stream monitoring, and more.

For details about using the Administration API, see "Using the Administration API" in the *Adobe Flash Media Server Configuration and Administration Guide*. It is also described in detail in the *Adobe Flash Media Server Administration API Reference*.

Deploying Flash Media Server 3

Installing Flash Media Server 3 is a simple process, whether deploying to Linux or Windows platforms. Your first step is to design your deployment structure. Flash Media Server can consist of individual installations, a single publishing point connected with a CDN, or a more complex Edge/Origin architecture, if required. It is recommended that you consult the *Adobe Flash Media Server Technical Overview* manual to help you assess your needs and design your specific deployment.

Regardless of your deployment structure, you must run the installer on each computer on which you want to run Flash Media Server. Once it is installed, you need to configure each server individually, designating Edge and Origin servers, and so on. For detailed installation instructions, refer to the *Adobe Flash Media Server Installation Guide*.

Upgrading from Flash Media Server 2

All of your Macromedia Flash Media Server 2 and Macromedia Flash Communication Server applications will be fully compatible with Flash Media Interactive Server 3, so upgrading should be a seamless process. No upgrade is available from Flash Media Server 2 to Flash Media Streaming Server 3 because it is a new product. To upgrade to Flash Media Streaming Server, you must purchase a full license.

First, the target installation folder for Flash Media Server 3 is different from the Flash Media Server 2 installation folder. In Windows, the default folder is `Program Files\Adobe\Flash Media Server 3`. In Linux, the default folder is now `/opt/adobe/fms`.

Configuration files from Flash Media Server 2 are *not* compatible and you should be sure to back up all your configuration files before upgrading. You will need to transfer your configuration manually to the new server. Server-side ActionScript and client-side ActionScript are fully compatible with Flash Media Server 3.

Flash Media Server 2 components for Flash will continue to be supported with the next version, but the components have been discontinued and will not be updated in future versions.

The Flash Media Server 2 Management Console has been renamed "Flash Media Administration Console." It has not changed significantly; the debug does support H.264 playback.

For more detailed upgrade instructions, refer to the *Adobe Flash Media Server Installation Guide*.

Verifying installation

After installing Flash Media Server on your server, you'll want to confirm that it was installed correctly. First connect to the server using the Administration Console.

`Start > Programs > Adobe > Flash Media Server > Administration Console` (Windows)

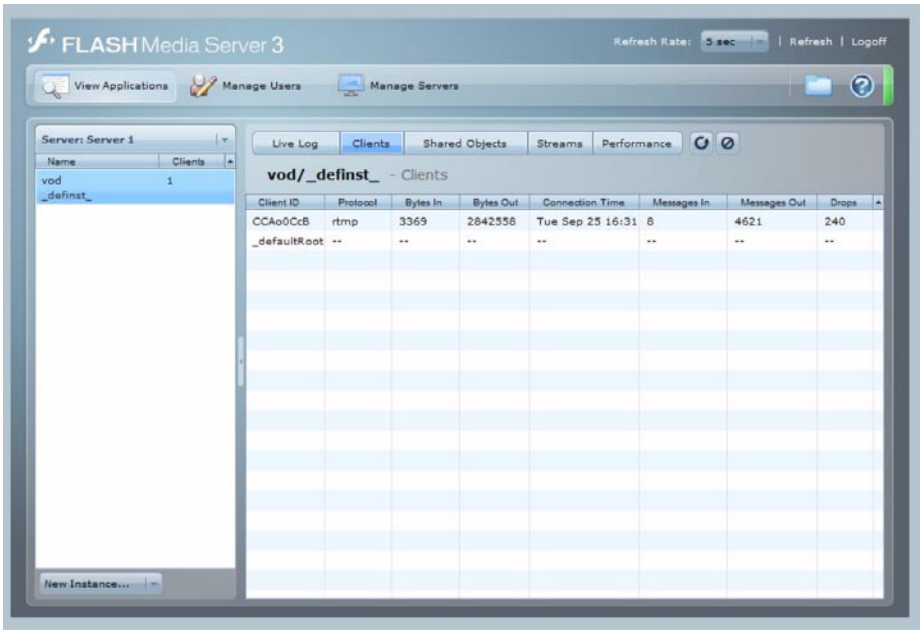
`opt/adobe/fms/fms_adminConsole.htm` (Linux)

The console can help you verify which applications can run. Try starting the new vod or live applications. After logging in, click Video Applications, then click New Instance and select "vod" or "live." If your server is working, you should see them start in the panel above (see the following figure).

You can also test your installation by running the vod sample application located in `samples/applications/vod/vodtest.html`. You can select a video (FLV or H.264) to play.

The *Adobe Flash Media Server Installation Guide* also provides a list of installed files and their locations. If you are having trouble getting Flash Media Server to run, you may want to consult this list to be sure your installation was complete.

Flash Media Server Administration Console



Configuring adaptors and virtual hosts

Enhancement in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

The server is divided into hierarchical levels: server, adaptor, virtual host, and application. The server is at the top level and contains one or more adaptors. Each adaptor contains one or more virtual hosts, and in turn, each adaptor contains one or more applications, or services. You can add adaptors and virtual hosts to configure the server for hosting multiple applications and sites.

If you are hosting multiple websites on a server, use separate virtual hosts to give customers their own root folders. This will allow you to keep separate settings, content, and log data for each customer.

You can assign an IP address or port number to an adaptor, but not to a virtual host. So, for example, if your customer needs their own SSL certificate, you'd want to assign their virtual host to its own adaptor.

Optimizing server performance

There are several levels of configuration available with Flash Media Server:

- Server
- Adaptor
- Virtual host
- Application

Note: All editions of Flash Media Server 3 are preconfigured and ready to stream, right out of the box. Changing settings may affect the performance and reliability of the server, so make changes carefully and be sure to backup the original XML files before making any changes. The most commonly edited settings are found in the FMS.ini file.

Each level has its configuration settings stored as XML files in the `RootInstall/conf` directory. There are also separate configuration files in this directory that control administrator accounts and logging.

Configuration files can be edited in any text or XML editor. Remember, for any changes to take effect, you must restart Flash Media Server. For changes to the administrator account settings (`Users.xml`), you must also restart the Flash Media Administration server.

Server-level settings

Enhancement in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

The server has one initialization settings file, `fms.ini`, in the `RootInstall/conf` directory. This file contains commonly used settings including the administrator username and password, and the settings you chose during installation. There is also only one `Server.xml` file, which controls settings such as connection request limits, idle connection timeout, IPv6 setup, SWF file verification, allowable domains, SSL/RTMPE configuration, and logging preferences. Edits made in the `Server.xml` file affect the entire server, unless they are overridden in a subsequent configuration file.

Adaptor settings

Enhancement in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

The `Adaptor.xml` file is the configuration file for individual network adaptors. It determines settings such as the number of threads that can be used by the adaptor, the communications ports that the adaptor binds to, and the IP addresses or domains from which the adaptor can accept connections and RTMP versions that can be used. You can also implement SSL with the `Adaptor.xml` file, if you want each of your adaptors to use a different digital certificate.

Each adaptor has its own directory inside the `RootInstall/conf` directory. The name of the directory is the name of the adaptor. Each adaptor directory must contain an `Adaptor.xml` file.

For example, the default adaptor included with the server at installation is named `_defaultRoot_`, and its directory is `conf/_defaultRoot_`. To change an adaptor's settings, edit the elements in its `Adaptor.xml` file.

Virtual host settings

Enhancement in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

Setting up your server with virtual hosts allows you to maintain distinct separation between hosting accounts on the server for Flash Media Server. Each virtual host directory on the server contains its own `Vhost.xml` file. The `Vhost.xml` configuration file defines settings for an individual virtual host. These settings include aliases for the virtual host, the location of the virtual host's application directory, limits on the resources the virtual host can use, and other parameters.

Each virtual host must have its own directory inside the adaptor directory. The name of the directory must be the actual name of the virtual host, such as `streaming.adobe.com`. Each defined virtual host must be mapped to a Domain Name Server (DNS) entry or another name resolution such as a Windows Internet Name Service (WINS) address or a hosts file, that specifies an IP address on the server computer.

Each adaptor must contain a `_defaultVHost_` directory in addition to the custom virtual hosts that you define. If a client application tries to connect to a virtual host that does not exist, the server attempts to connect it to `_defaultVHost_`. (If you are using a secure port for the adaptor that contains the virtual host, you can only define one virtual host for the adaptor, in addition to `_defaultVHost_`.)

Application settings

Enhancement in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

The `Application.xml` file contains the settings for Flash Media Interactive Server applications. These include settings such as: the size of the Server-Side Media ActionScript runtime engine, the location at which streams and shared objects are stored, default AMF encoding, and bandwidth limitations.

The `Application.xml` file in the virtual host directory configures the default settings for all applications within that virtual host. If you want to have different settings for a particular application, you can copy an Application.xml file to the application's registered application directory (`/applications/app_name`) and edit it to include your custom settings.

In most cases, the settings in `Application.xml` file in the specific application directory override settings in the `Application.xml` file in the virtual host directory, but not always. For more details, see the *Adobe Flash Media Server Configuration and Administration Guide*.

User settings

Enhancement in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

To add, remove, or set permissions for Flash Media Server administrator accounts, edit the `Users.xml` file in the root level of the configuration directory. You can also configure Server Management API calls to the Flash Media Administration Server (for example, to allow/deny access to specific HTTP calls) within this settings file.

Log settings

Enhancement in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

With Flash Media Server 3, you have powerful logging capabilities, which can be highly customized for your specific application.

Also located at the root level of the configuration directory, the `Logger.xml` file controls settings for Flash Media Server log files. You can edit this file to specify the data that is logged, where the log files are saved, and how often they are rotated. The default location for the log files is in the logs directory in your server installation directory (`RootInstall/logs`).

The Logging section in `Server.xml` enables or disables the log files; `Logger.xml` contains the actual log file settings.

Note: Log files and field names within the log files are written in English. Some content in the log file, however, may be in another language, depending on the operating system.

Configuring performance features

Even with all of the built-in improvements in Flash Media Server 3, you may want to customize your installation for maximum efficiency. There are also several features that you may want to optimize for your specific needs. They are described here; for more detailed information, see the *Adobe Flash Media Server Configuration and Administration Guide*.

Note: For more detailed tips for optimizing Flash Media Server specifically for Linux installations, consult the article "Performance-tuning Flash Media Server 2 for live webcasts using Linux," at *www.adobe.com/devnet/flashmediaserver/articles/performance_tuning_webcasts.html*.

Stream cache/stream chunks

Enhancement in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

When a stream is requested from the server, segments of the stream are stored in a cache on the server. As long as the cache has not reached capacity, the server places segments in the cache. When the cache is full, the server removes unused segments, starting with the least recently used. You can set the size of the cache by changing the `SERVER.FLVCACHE_MAXSIZE` parameter in the server's `fms.ini` file. The default value is 500MB.

Streamed content is broken into chunks as it is sent via RTMP. The size of these chunks can also be set in the `fms.ini` file. The `APP.DEFAULT_CHUNKSIZE` parameter can be between 128 and 1024B, with a default value of 128. (You can also set the stream chunk size specifically for the VOD service separately by setting the `APP.VOD_CHUNKSIZE` parameter.) Larger values will reduce CPU usage, but can also slow performance for clients on lower bandwidth connections.

When setting the size of the cache and stream chunks, it is important to note that setting the size too high can actually result in slower performance. For example, if the cache size is greater than the available memory, or the server process exceeds the 2GB OS limit, the server process could be terminated. Alternatively, if it is set too low, all segments could theoretically be in use and unable to be exchanged for new stream segments. In this case, the stream requesting the new segment will stop playing.

Process scopes

Enhancement in Flash Media Interactive Server; requires Flash Player 6 or later.

Flash Media Interactive Server can be scoped into different processes to increase the capacity of the server by overcoming natural OS limits for CPU/RAM and for process isolation. Splitting your processes:

- Allows Flash Media Server to accept connections faster
- Allows Flash Media Server to store more FLV/MP3 data in RAM
- Expands the 2GB memory limit
- Isolates the instance/application/VHost or adaptor from malformed scripts or denial of service (DoS) attacks.

For more information on process scopes, see the *Managing Flash Media Server* manual.

Flash Media Interactive Server lets you further increase your capacity and quality of service using distributed process scopes. Distributed process scopes and the `distribute` setting will be discussed in the next section.

You can configure Flash Media Server to spawn FMSCore processes by configuring the scope node in the global `Application.xml` file. Valid scopes include `adaptor`, `vhost`, `app`, `inst`. The adaptor scope is a new feature in Flash Media Server 3.

Depending on the scope you choose, you can configure each core process separately. For example, if you configure the system to scope adaptors to different core processes, each setting in the specific `adaptor.xml` file and all subsequent xml files would be used to configure that core process. In this configuration, you could set each core process to listen on a different port, or change the SSL or HTTP tunneling settings.

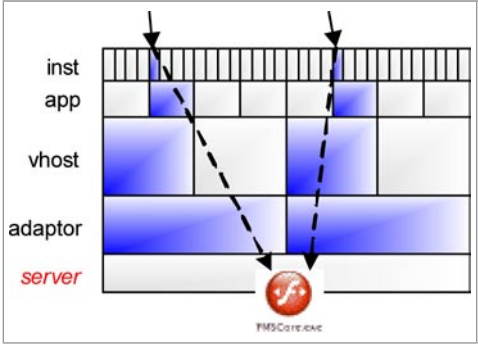
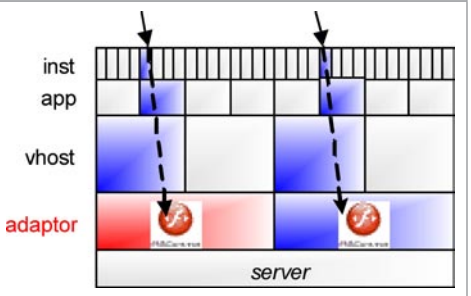
In another example, if you set scope to "app," each core process could be configured with bandwidth limits—provided you had a separate `application.xml` file for each application running on your system.

The following process configuration is the default.

```
<Process>
    <Scope></Scope>
    <Distribute numprocs="1"></Distribute>
    <LifeTime>
        <RollOver></RollOver>
        <MaxCores></MaxCores>
    </LifeTime>
    <MaxFailures>2</MaxFailures>
    <RecoveryTime>300</RecoveryTime>
</Process>
```

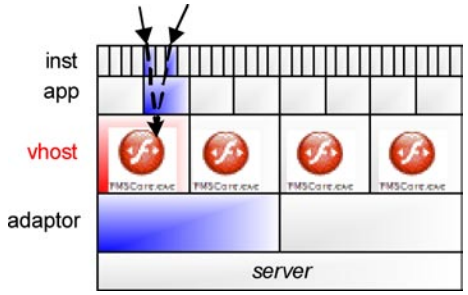
Let's look at how changing the scope setting affects your system. The following figures illustrate the effect of changing your scope configuration.

All of these settings are optimal for stateful applications where clients need to communicate with each other (chat, live video, gaming, or data sharing solutions).

<scope></scope>	**<scope>adaptor</scope>**
No process scope (default setting). All clients connect to a single scope.	Process scope "adaptor." Each adaptor has its own core process. Virtual hosts and applications running on different adaptors will not interfere.
	

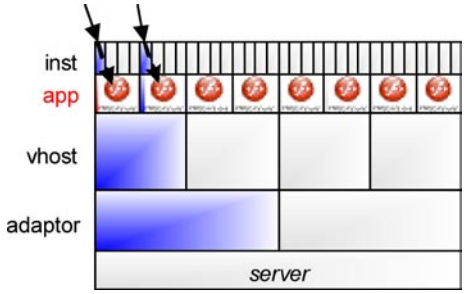
<scope>vhost</scope>

Process scope "vhost." Useful for applying unique settings for users in different subdomains.



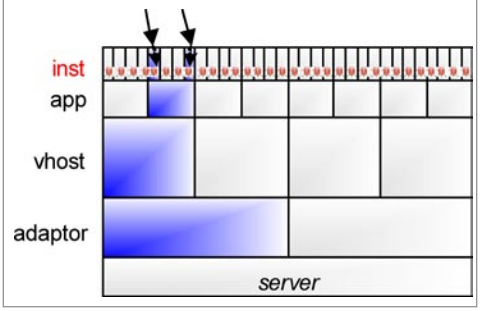
<scope>app</scope>

Process scope "app." Each application running can be run within its own process. This would be useful if you have lots of memory, and heavy connection requests from different applications.



<scope>inst</scope>

Process scope "inst." Depends on your customer provision.



For a CDN customer (VOD profiles), selection of a scope depends upon how you provision your customer accounts. Consider that each customer is provisioned as an application within Flash Media Server. If there are a lot of customers, the app scope may not be the best option. Each process scope can use a maximum of 4GB of RAM, so too many processes may not be the best choice. Alternatively, if you have only a few large customers in your deployment, then the app scope may be your best choice.

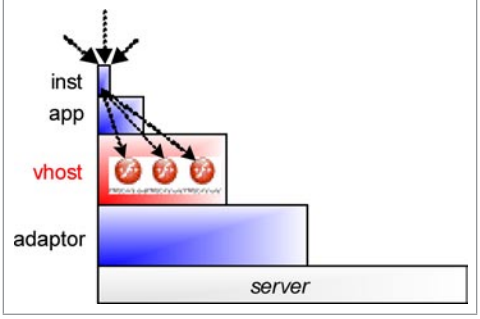
Distributed cores

New feature in Flash Media Interactive Server; requires Flash Player 6 or later.

To further increase the capacity and reliability of your server, you can distribute connections across multiple processes for a specific scope. For example, if your scope was set to adaptor, you could have connections spread across any number of core processes for each virtual host. In the following figure, there are four core processes for each virtual host. All connections on the single virtual host could be evenly distributed over three processes. If there were two virtual hosts, connections could be distributed across six core processes.

Distributed process scopes

There are three processes, each able to accept connections. Users connected to the same application may not be able to communicate or share RAM because they connected to a different process. Useful for stateless deployments.



Like process scopes, the distributed core feature lets you increase the capacity of your server. Distributed cores will let you engage more RAM for the cache and more threading for the process-intensive connection routine. Distributed cores are best used for VOD applications (commercial or social). They cannot be used when deploying a multiway hybrid or live solution because connections need to be on the same core process to share communication.

Inside the `Application.xml` configuration file, the default settings disable distributed process scopes.

```
<Process>
    <Scope></Scope>
    <Distribute numprocs="0"></Distribute>
    <LifeTime>
        <RollOver></RollOver>
        <MaxCores></MaxCores>
    </LifeTime>
</Process>
```

You can distribute connections to scopes in `<Scope>`. The following figure shows your options for process distributions. The left column shows the settings for `<Scope>` and the top row is the settings for `<Distribute>`. As you can see, if you set `<Scope>vhost</Scope>`, your options for `<Distribute>` are app, inst, and clients.

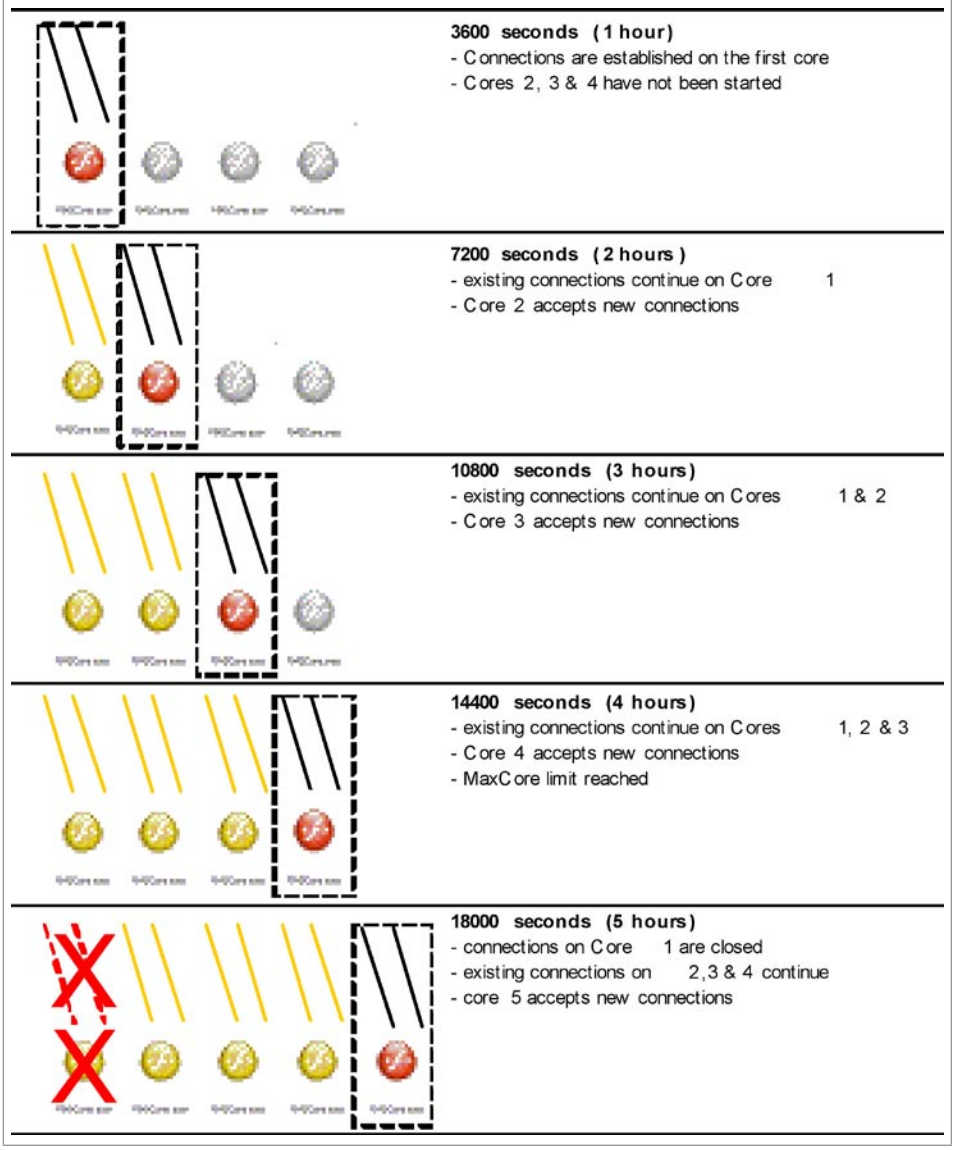
Distributed process scope options

	Distributed Processes				
	adaptor	vhost	app	inst	clients
server	☑	☑	☑	☑	☑
adaptor		☑	☑	☑	☑
vhost			☑	☑	☑
app				☑	☑
inst					☑

The following example shows process distribution with a single virtual host. The figure shows how the configuration would be implemented over a 5-hour time frame given a 1-hour (3,600 second) rollover.

```
<Process>
    <Scope>adaptor</Scope>
    <Distribute numprocs="1">vhost</Distribute>
    <LifeTime>
        <RollOver>3600</RollOver>
        <MaxCores>4</MaxCores>
    </LifeTime>
</Process>
```

Distributed process scope options

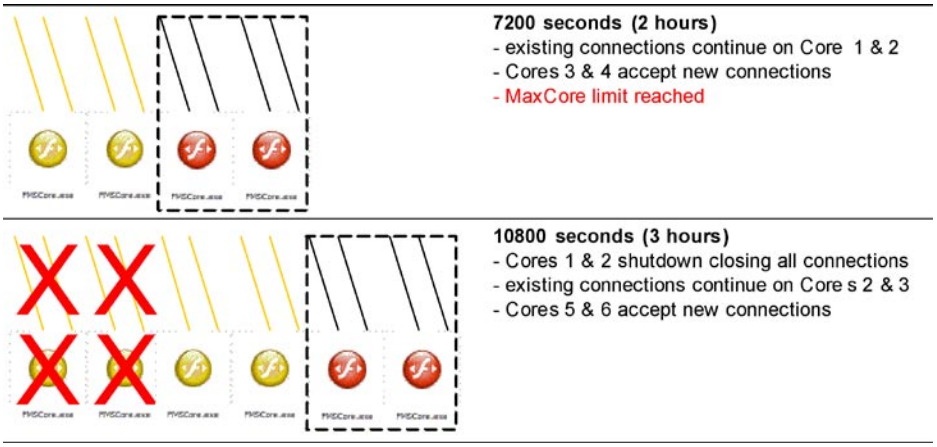

3600 seconds (1 hour)
- Connections are established on the first core
- Cores 2, 3 & 4 have not been started
7200 seconds (2 hours)
- existing connections continue on Core 1
- Core 2 accepts new connections
10800 seconds (3 hours)
- existing connections continue on Cores 1 & 2
- Core 3 accepts new connections
14400 seconds (4 hours)
- existing connections continue on Cores 1, 2 & 3
- Core 4 accepts new connections
- MaxCore limit reached
18000 seconds (5 hours)
- connections on Core 1 are closed
- existing connections on 2,3 & 4 continue
- core 5 accepts new connections

The maximum connection time using the above configuration is 4 hours. After 4 hours the core process will be closed and all connections will be dropped. At this stage, the client SWF should initiate a reconnect routine that will reestablish the stream playback.

Changing the number of processes (numprocs) configures the number of processes in which connections are distributed. Changing the numprocs setting to "2" given the same settings as before will reduce the maximum connection time from 4 hours to 2 hours. After the second hour, the `MaxCores` limit of 4 will be reached. To spawn two more processes, the first two processes will be killed and connections will be closed.

```
<Process>
 <Scope>adaptor</Scope>
 <Distribute numprocs="2">vhost</Distribute>
 <LifeTime>
   <RollOver>3600</RollOver>
   <MaxCores>4</MaxCores>
 </LifeTime>
</Process>
```

Example of distributed process configuration


7200 seconds (2 hours)
- existing connections continue on Core 1 & 2
- Cores 3 & 4 accept new connections
- MaxCore limit reached
10800 seconds (3 hours)
- Cores 1 & 2 shutdown closing all connections
- existing connections continue on Core s 2 & 3
- Cores 5 & 6 accept new connections

The maximum lifetime for connections in this configuration is 2 hours because the `MaxCore` limit is set to 4 and the `numProcs` is set to 2.

To calculate the *maximum connection time* for clients, use this formula:

```
(MaxCore DIV NumProcs) x Rollover
```

```
Example: (4 DIV 2) X 3600 = 7200 seconds (2 hours)
```

To calculate the total number of processes, use this formula:

```
Number of Scopes X NumProcs X MaxCores
```

Example using `<Distribute>adaptor</Distribute>` with two adaptors:

```
2 X 2 X 4 = 16 Core processes
```

To calculate the amount of RAM required to support the distributed processes, multiply the total processes by 4. Each core process can use up to 4GB of RAM; therefore the previous example with 16 cores could consume up to 64GB of RAM.

Close idle connections

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

When clients leave an application, their connection is often left open. Flash Media Server 3 can now detect these idle connections and reclaim their resources for new and active clients. When a client has been idle longer than the maximum specified idle time, (the default is 10 minutes), the server will close the connection. To enable this feature, you must set `AutoCloseIdle-Clients` to true in the `Server.xml` file. Once it is enabled in the `Server.xml` file, you can disable it for individual virtual hosts or individual applications in the `Vhost.xml` and `Application.xml` files.

Limit connection requests

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

In some cases, if many clients are attempting to connect to the server, the quality of service can be diminished for those clients who are already connected. By setting the `MaxConnection Rate` in the `Server.xml` file, you can limit the number of connection requests per second that are accepted.

Send aggregate messages

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 9,0,115,0 or later.

By default, applications break up aggregate messages into individual messages before delivering them to clients. By default, aggregate messages are enabled. Applications can be configured not to deliver aggregate messages by setting the `AggregateMessages` parameter to false in the `Application.xml` file.

Sending aggregate messages reduces CPU usage and increases server capacity. However, it can introduce some latency, so it is not recommended for real-time one-to-one communication. It is recommended for on-demand and live broadcast applications where latency will have little effect.

Configure content storage

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

Storage configuration is important to server performance. If your hard disk access is not fast enough to keep up with the bitrate of your content, your clients will receive buffer empty messages and the overall quality of service will suffer.

The server can use local or network storage to serve media files. In the `Application.xml` file, you can change the default location where streams and shared objects are stored, and map virtual directories to physical directories on local or network storage to manage your content. This can be very convenient in team environments, when you may not always want content creators to have direct access to your Flash Media Server, or if you have a large library of media files that you don't want to copy to your Flash Media Server.

Configuring security features

Flash Media Server 3 has several security features that you can easily set in the configuration files. (For more information about securing your server and content, refer to the "Security features" section.)

Verify SWF files

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 9,0,115,0 or later.

The server can be configured to verify client SWF files before allowing them to connect to an application. Verifying SWF files prevents someone from creating their own SWF files that attempt to stream your content, or using your server resources. The server compares the connecting SWF with existing SWF files on your Flash Media Server, and allows the connection if there is a match.

In the `Application.xml` file, you can specify one or more folders on the server to hold these SWF verification files. This is done within the node, `<SWFVerification>`. You can also configure the versions to check the length of time the verification data is held in cache, and any exceptions (such as Flash Media Encoder). You can also configure how often the server should check for updated SWF verification files. Here is a sample from the `Application.xml` file. By default `SWFVerification` is turned off.

```
<SWFVerification enabled="true">
    <SWFFolder />
    <MinGoodVersion />
    <UserAgentExceptions>
        <Exception from="" to="" />
    </UserAgentExceptions>
    <Cache>
        <TTL>1440</TTL>
        <UpdateInterval>5</UpdateInterval>
    </Cache>
</SWFVerification>
```

Allow domains to connect to a virtual host

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

You can specify a list of domains that are allowed to connect to a particular virtual host. By default, connections are allowed from all domains. Set the `VHOST.ALLOW` parameter in the `fms.ini` file to a comma-delimited list of host names, domain names, and full or partial IP addresses you wish to allow access to the virtual host.

Limit access to Flash Media Administration Server

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

By default, a client can connect to Flash Media Administration Server from any domain or IP address, which can be a security risk. You can change this by editing the `AdminServer` parameter in the `Server.xml` file. Simply add a comma-delimited list of host names, domain names, and full or partial IP addresses you want to allow. The default value is "all."

Administration API via HTTP

Feature available in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

You can monitor Flash Media Server via simple HTTP commands. The Flash Media Administration API methods can be called over HTTP connections. In the `fms.ini` file, set the `USERS.HTTPCOMMAND_ALLOW` parameter to a comma-delimited list of APIs. You can also specify methods on a more granular user-based level, in the `Users.xml` file.

Most users leverage this feature to write custom monitoring applications. You can monitor the capacity of the server including the number of connections, bandwidth used, number of streams in cache, and even restart the server.

Encrypted RTMP (RTMPE)

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 9,0,115,0 or later.

By default, the encrypted RTMP (RTMPE) is enabled in the server's `Adaptor.xml` file, and offers 128-bit encryption. If you wish to disable RTMPE, simply change the `ADAPTOR.RTMPE_ENABLED` parameter to "off."

You should only turn off encrypted RTMP if you do not want it to be available to developers deploying applications on your server. (RTMPE does require more CPU power compared to standard RTMP). Otherwise, you should leave it enabled so it is available when you want to use it.

Secure Sockets Layer (SSL)

Feature available in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

Like RTMPE, SSL is a protocol that enables more secure communication. Unlike RTMPE, SSL requires a certificate signed by an intermediate Certificate Authority, and requires configuration to enable. SSL must first be configured in the `Server.xml` file; certificates can also be set up to secure independent adaptors or independent virtual hosts.

For more details about security settings for Flash Media Server, refer to the "Securing content with Flash Media Server 3" section.

Configuring general settings

There are a number of helpful settings, that don't necessarily directly affect security or performance, but can streamline and customize your application deployment.

Allow application debugging connections

Feature available in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

The Flash Media Administration Console and the API can actually connect and "listen" to SharedObjects and Streams. This feature makes it easier to debug complex server-side applications. By default, the server does not allow debugging connections. However, the Flash Media Server Administration Console must make this special debugging connection to the server to play back streams and access shared object data.

To allow debugging connections, edit the `Application.xml` file of the virtual host or application you'd like to access. You will edit the node shown below to enable debug connections. After setting, you will need to restart both the Flash Media Server and Flash Media Server Administration service and reload the Administration Console.

```
<Debug>
    <MaxPendingDebugConnections>50</MaxPendingDebugConnections>
    <AllowDebugDefault>true</AllowDebugDefault>
</Debug>
```

Alternatively, you can use HTTP to monitor the server activity.

Define application object properties

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

With Flash Media Server 3, you can define global properties that are accessible to all application instances on a specific virtual host. Alternatively, if you define these application properties in an `Application.xml` folder in a particular application folder, the properties are only available for that application.

For example, you could create a global `application_owner` property, and every instance of that application could access that property using this syntax:

```
Application.config.application_owner
```

Enable/disable native bandwidth detection

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

Flash Media Server 3 can detect a client's bandwidth using native bandwidth detection, or in server-side ActionScript (called script-based bandwidth detection). Because native bandwidth detection is built into the core server code, it is much faster than script-based detection. Native bandwidth detection is enabled by default, and can be further configured in the `Application.xml` file.

Activity logs

Feature available in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

Flash Media Server 3 offers real-time server monitoring and extensive logging capabilities to help you with server management and troubleshooting. The log files track activity such as general traffic and server load, who is accessing the server, client behavior and interaction, and general diagnostics.

Flash Media Server maintains several different types of logs:

- Access logs—Tracks information about users accessing the server
- Application logs—Tracks information about activities in application instances
- Diagnostic logs—Tracks information about server operations

Flash Media Server access log files are written in W3C format. You can use standard parsing tools to parse these log files.

Access logs

The access log records information about connection requests by Flash Player and Flash Media Server application instances. The default configuration creates a single access log per server, called `access.XX.log`, which is located in the Flash Media Server logs directory. (You can also configure Flash Media Server to create a separate access log for each virtual host.) The "XX" in the filename is a two-digit number representing the version of the log (for example, `access.00.log` would contain the most recent logs).

The access log will record data such as:

- Date and time a client connected to the server
- How much total bandwidth was consumed during the session
- Which streams were accessed by the connection
- Whether the client published a stream
- Whether the client jumped to a new location within a recorded stream

Application logs

The application logs record information about activities in application instances. These logs are especially useful for debugging applications.

The default configuration creates one application log per application instance, called `application.XX.log`, which can be found in the application/instance folder of the corresponding virtual host.

Application logs record application event data such as:

- Date and time of the event
- An event's server process ID
- Event status level (warning, error, information, debug, and so on)

Diagnostic logs

The diagnostic logs record information about Flash Media Server operations and are used primarily for debugging server-level issues.

Flash Media Server is configured, by default, to create a diagnostic log for each type of process. The default diagnostic logs are `master.XX.log`, `edge.XX.log`, `core.XX.log`, `admin.XX.log`, and `httpcache.XX.log`. All of the diagnostic logs are located in the Flash Media Server logs directory.

A wide variety of useful data such as information about stream events, application instances, virtual hosts, and Edge/Origin issues, and more, can be obtained through close examination of the diagnostic log files.

Using server tools

Administration API

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

The Administration API can be used to monitor, manage, and configure the server from a Flash Player or Adobe AIR client over RTMP or from a web client over HTTP. The Flash Media Server Administration Console was built using the Administration API. You can also use the API to create custom administration tools. Some of the methods available to you include:

- Add/delete administrative users
- Start and stop the server, virtual hosts, and applications
- Initiate garbage collection
- Get and set server configuration

The API is described in detail in the *Adobe Flash Media Server Administration API Reference.*

Server healthcheck utility

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Windows/Linux; no Flash Player.

In addition to the Administration Console and server logs, there is a new tool available to assist you in evaluating the overall health of Flash Media Server. `FMSCheck.exe` is a command line utility program that can be used to diagnose and determine server status. The tool is installed with Flash Media Server and is available for both Windows and Linux.

As a command-line utility, it can easily be integrated with your back-end monitoring systems. `FMSCheck` provides information such as whether the server is running or not, what the response time is, and which core processes or applications may not be responding. It can also check every active instance of every application currently running on the server to be sure each one is accepting connections as expected.

This tool can save the Flash Media Server administrator the significant time and effort of manually checking applications and streams. Test connections can be run in parallel or staggered, depending on the desired test.

Video validation utility

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Windows/Linux; no Flash Player.

With the large number of encoding technologies able to convert video into a Flash-compatible format, inconsistencies in video encoding can occur. This new tool, `FLVCheck.exe`, will let you validate if a particular video can be streamed from Flash Media Server. If the file has been corrupted or encoded with missing information, this tool will let you know.

FLVCheck will validate both FLV and MPEG-4 file formats. It will check the data structure including the headers, time stamps, and metadata. It is a command line tool that can be scripted or used in an automated environment. For FLV files (On2 and Sorenson codecs), the tool will even fix minor errors to make sure the files run or that they do not corrupt the server.

Scaling Flash Media Server 3

Servers have a finite capacity, so as traffic and throughput increases, applications need to be scaled to preserve quality of service. Flash Media Server offers several flexible options for graceful scaling of high-traffic applications.

Cluster deployment

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

You can deploy multiple servers behind a load balancer to distribute the application load evenly. Flash Media Server clustering enables you to scale an application to accommodate more clients reliably, and creates redundancy, which eliminates single points of failure. This approach is generally best for live or VOD streaming, where clients do not need to communicate with each other from within specific application instances. Clustering can be achieved using either Flash Media Streaming Server or Flash Media Interactive Server.

Flash Media Server intelligent balancing

New feature in Flash Media Interactive Server; requires Flash Player 6 or later.

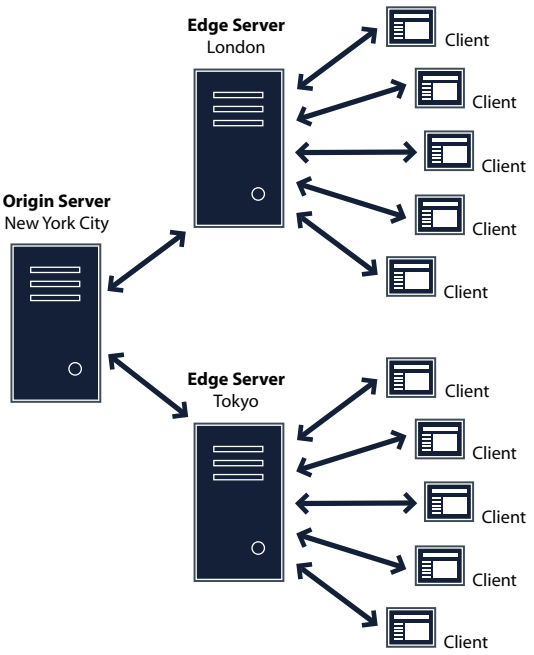
With the Flash Media Interactive Server, you can intelligently direct traffic to a multiple server cluster using server-side scripting. This option would typically be used for multiway communication applications that require connections be routed to a specific server. This option does require development of rather sophisticated server-side ActionScript to manage connections.

Edge/Origin configurations

New feature in Flash Media Interactive Server; requires Flash Player 6 or later.

In the past, Flash Media Server distributed caching/load balancing was achieved by purchasing the Edge/Origin editions. This functionality is now built into Flash Media Interactive Server. Flash Media Interactive Server provides an enterprise-ready architecture designed to simplify load balancing, failover, and clustering to ensure maximum availability over large regions. The following figure shows the Edge/Origin architecture.

Edge/Origin architecture



How Edge/Origin deployments work

Edge/Origin server configurations improve performance by distributing the server load among many computers on a network. With an Edge/Origin deployment strategy, all connection requests from clients are redirected to an Edge server. The configuration also lets you maximize your network if you are supporting a large local network. By placing Edge servers in remote office locations, the Edge servers will cache media files locally so each stream does not need to access the Origin (host) server for each stream.

Typically Edge/Origin deployments are best used with one-way streaming services. When using custom server-side applications to enable real-time communication, the Edge server strictly handles the requests on behalf of the Origin server. Client connections then make round-trips to the Origin server to run the application.

In Flash Media Interactive Server 3, Edge-level support for bandwidth detection and stream length detection has been integrated. The first server in the chain (Edge or Origin) receiving a stream call will also handle the bandwidth check and stream length check without calling the Origin server script layer. This feature is compatible with the FLVPlayback component for Flash 8 or Flash CS3 Professional.

When a client request is received, the Edge server will handle the tasks it can, then will make a connection to the Origin server for any additional data required. When the Origin server fulfills the request, the data is sent back to the Edge server, then on to the client. To the client, it appears that the connection is made directly to the application running on the Origin server.

The Edge server serves as a "traffic cop"—handling connection overhead, authentication, and other administrative duties—freeing up valuable system and network resources for the Origin server. Every connection and connection attempt consumes resources over and above the actual stream data flowing through the connection. As the number and frequency of connections increase, the load can be excessive; adversely affecting server performance. The Edge server greatly reduces this load by aggregating connections. The Edge multiplexes the connections from a large number of clients into one connection to the Origin server. All communications between Edge and Origin servers are transparent to clients.

The Edge server also stores the prerecorded media content received from the Origin server in a cache, which is then made available to other clients that connect to the Edge server. Caching static content further reduces the load on the Origin server.

Deployment strategies

A simple way to distribute load among Edge servers is to assign users in a geographical region or other delineation to a specific Edge server. For example, one Edge server may aggregate and forward requests from clients in London, while another may handle requests from Tokyo.

A typical networked Flash Media Server deployment can involve multiple Edge servers, deployed either individually or in clusters. Edge servers can also be chained, allowing even further distribution of traffic.

To enable the Edge/Origin feature, you can configure any server in your cluster as your Origin server (or servers), and the rest as your Edge servers. All editions in an Edge/Origin configuration must be the same (for example, you cannot mix Flash Media Streaming Server and Flash Media Interactive Server editions in a cluster).

Deployment

Large-scale Flash Media Server deployments are supported with the Flash Media Server Edge/Origin configuration. For an introduction to Flash Media Server Edge/Origin, please refer to the *Using Flash Media Server Edge Servers* manual. For instructions for setting up an Edge server, refer to the "Installing Edge servers" section in the *Installing Flash Media Server 2* manual.

Edge servers are also referred to as proxy servers. There are four ways to configure a Flash Media Server Edge (or proxy) server:

- Client auto-discovery proxy
- Server auto-discovery proxy (reverse proxy)
- Explicit URI
- Implicit URI (recommended)

Typically, implicit URI is the recommended setting because it is the most secure and requires the least amount of communication. It can hide the Origin server URI and it is the easiest to set up. Therefore, we will explore the implicit URI configuration. All of these methods are described in the *Using Flash Media Server Edge Servers* manual.

Configuring proxy (Edge) servers using implicit URI

In typical large-scale deployments, your business could deploy the implicit URI method.

The following settings define the virtual host as a proxy (Edge) server.

```
<Proxy>
    <Mode>remote</Mode>
    <Anonymous>false</Anonymous>
    <CacheDir enabled="true" useAppName="true">d:\fmsCache\</
    CacheDir>
    <LocalAddress></LocalAddress>
    <RouteTable protocol="">
        <RouteEntry>edge1.fms.com:*;192.168.110.150:1935</RouteEntry>
    </RouteTable>
</Proxy>
```

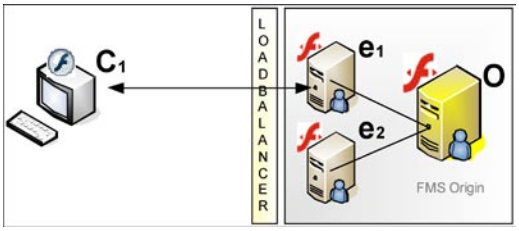
This configuration will allow the client to connect with the Edge Server without exposing the proxy server.

The connection string would look like:

```
rtmp://edge1.fms.com/ondemand/
```

The Flash Player would connect to the Edge server and not expose the Origin Server at 192.168.110.50, as shown in the following figure.

Edge server using a single Origin server



You can configure Edge servers to create proxy clusters. In the following figure, notice how an Edge server (e1) can proxy the Edge server (e0) in its routeEntry tag. The Edge server (e0) would proxy the Flash Media Server origin. This type of configuration will allow you build Flash Media Server Edge clusters that could be geographically balanced.

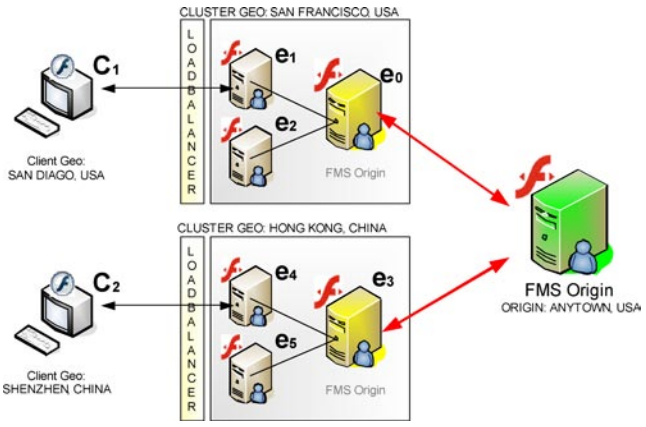
The `RouteEntry` for the cluster members would point to a main Edge server.

```
    <RouteTable protocol="">
      <RouteEntry>edge1.fms.com:*;edge0.fms.com:*;</RouteEntry>
    </RouteTable>
```

The `RouteEntry` for the main Proxy (Edge server) in the cluster would point back to the Origin server.

```
<RouteTable protocol="">
  <RouteEntry>edge0.fms.com:*;origin.fms.com:*;</RouteEntry>
</RouteTable>
```

Edge server cluster using a master Edge and a single Origin



Using live video

Flash Media Server allows you to broadcast live streams with a wealth of interactive features. Connected clients only need Flash Player or an Adobe AIR application to view the live broadcast.

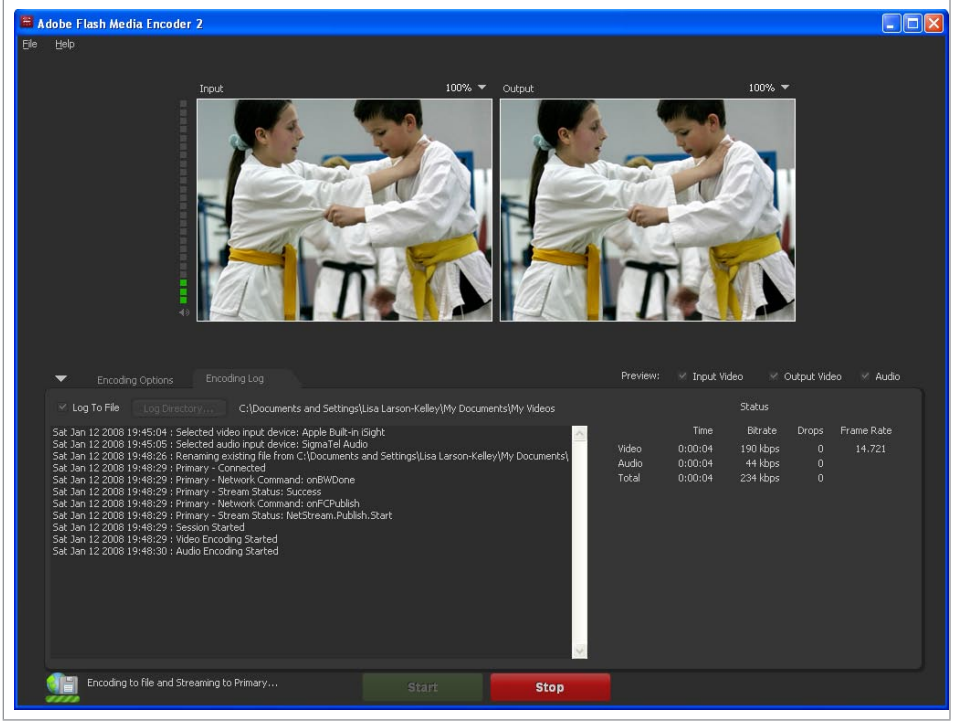
You can easily create your own custom live video broadcast application using the Flash integrated development environment (IDE) or Flex Builder. It's important to note that video captured and broadcast through a Flash Player interface will be encoded using the Sorenson Spark video codec and the Nellymoser audio codec. These formats are generally acceptable and quite efficient for real-time conferencing. However, if you require VP6 or MP3 encoding, or you don't need the flexibility of a completely customized encoding interface, you may want to consider using the standalone Adobe Flash Media Encoder.

Flash Media Encoder 2

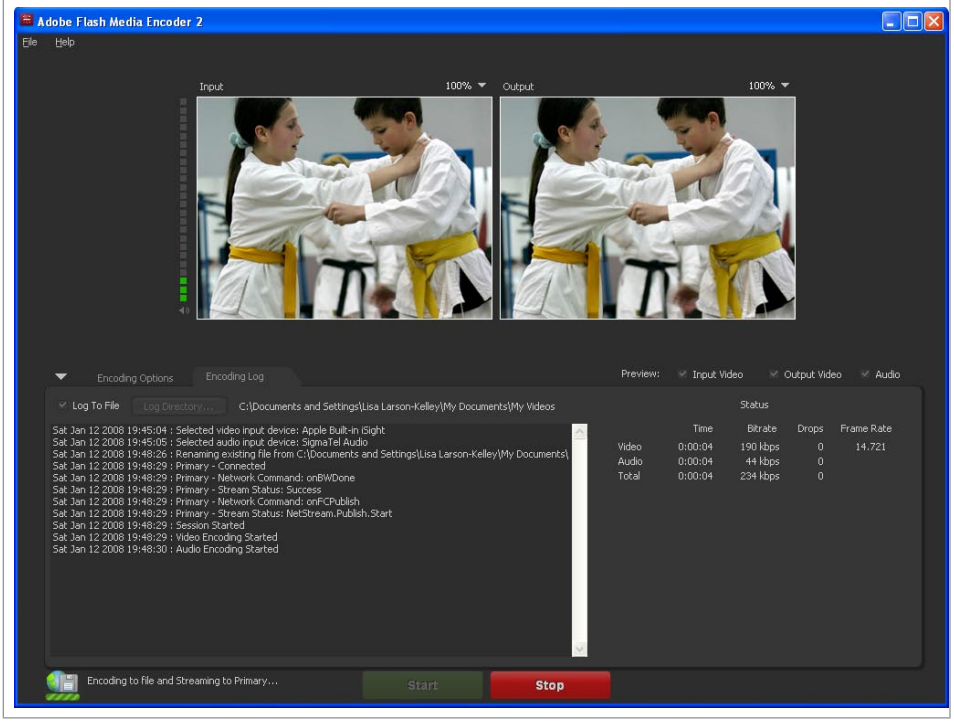
Adobe Flash Media Encoder 2 is a Windows Server, Windows XP, or Windows Vista® based live encoding application, designed to enable event producers to capture live audio and video while streaming it in real time to Flash Media Server or FVSS. Featuring simple out-of-the-box setup and operation, Flash Media Encoder requires no scripting knowledge to begin broadcasting high-quality 24x7 streams for events such as sporting events, concerts, webcasts, and seminars.

With an intuitive user interface (shown in the following figures) that works seamlessly with plug-and-play cameras, microphones, and compatible analog-to-digital converters, Flash Media Encoder 2 also provides size and bitrate flexibility. Flash Media Encoder 2 streams are compatible with Flash Player 8 or later.

Flash Media Encoder interface


Adobe Flash Media Encoder 2
File Help
Input 100%
Output 100%
Encoding Options
Encoding Log
Preview: Input Video Output Video Audio
Log To File
Log Directory...
C:\Documents and Settings\Lisa Larson-Kelley\My Documents\My Videos
Status
Time Bitrate Drops Frame Rate
Video 0:00:04 190 kbps 0 14.721
Audio 0:00:04 44 kbps 0
Total 0:00:04 234 kbps 0
Sat Jan 12 2008 19:45:04 : Selected video input device: Apple Built-in iSight
Sat Jan 12 2008 19:45:05 : Selected audio input device: SigmaTel Audio
Sat Jan 12 2008 19:48:26 : Renaming existing file from C:\Documents and Settings\Lisa Larson-Kelley\My Documents\
Sat Jan 12 2008 19:48:29 : Primary - Connected
Sat Jan 12 2008 19:48:29 : Primary - Network Command: onBWDone
Sat Jan 12 2008 19:48:29 : Primary - Stream Status: Success
Sat Jan 12 2008 19:48:29 : Primary - Network Command: onFCPublish
Sat Jan 12 2008 19:48:29 : Primary - Stream Status: NetStream.Publish.Start
Sat Jan 12 2008 19:48:29 : Session Started
Sat Jan 12 2008 19:48:29 : Video Encoding Started
Sat Jan 12 2008 19:48:30 : Audio Encoding Started
Encoding to file and Streaming to Primary...
Start
Stop

Flash Media Encoder live broadcast log


Adobe Flash Media Encoder 2
File Help
Input 100%
Output 100%
Encoding Options
Encoding Log
Preview: Input Video Output Video Audio
Log To File
Log Directory...
C:\Documents and Settings\Lisa Larson-Kelley\My Documents\My Videos
Status
Time Bitrate Drops Frame Rate
Video 0:00:04 190 kbps 0 14.721
Audio 0:00:04 44 kbps 0
Total 0:00:04 234 kbps 0
Sat Jan 12 2008 19:45:04 : Selected video input device: Apple Built-in iSight
Sat Jan 12 2008 19:45:05 : Selected audio input device: SigmaTel Audio
Sat Jan 12 2008 19:48:26 : Renaming existing file from C:\Documents and Settings\Lisa Larson-Kelley\My Documents\
Sat Jan 12 2008 19:48:29 : Primary - Connected
Sat Jan 12 2008 19:48:29 : Primary - Network Command: onBWDone
Sat Jan 12 2008 19:48:29 : Primary - Stream Status: Success
Sat Jan 12 2008 19:48:29 : Primary - Network Command: onFCPublish
Sat Jan 12 2008 19:48:29 : Primary - Stream Status: NetStream.Publish.Start
Sat Jan 12 2008 19:48:29 : Session Started
Sat Jan 12 2008 19:48:29 : Video Encoding Started
Sat Jan 12 2008 19:48:30 : Audio Encoding Started
Encoding to file and Streaming to Primary...
Start
Stop

Flash Media Encoder 2 can also be tightly integrated into your current streaming workflow with command-line control both locally and through a remote connection via Microsoft Remote Desktop Connection or Virtual Networking Computing (VNC). Among other powerful features, auto-restart after power failures or other system restarts also helps ensure that your live streams are reliably available. When using a device that can generate timecodes, you can also embed an SMPTE timecode into the video stream.

A significant benefit over broadcasting direct from the Flash Player, Flash Media Encoder 2 allows you to broadcast video using the VP6 codec. You also have greater audio control with bitrate-efficient Nellymoser or MP3 encoding.

Flash Media Encoder is available as a free download from Adobe: *www.adobe.com/go/flashmediaencoder.*

Data keyframes

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

With the new data keyframe feature in Flash Media Server 3, metadata can be sent whenever a new subscriber requests the stream. The metadata can also be updated at any time by dispatching a new data keyframe via server-side or client-side ActionScript, or via Flash Media Encoder. This new metadata is then received by all connected clients through the `onMetadata` event handler.

A data keyframe is a special data message that can be set to a live stream and stored in the memory of the server. Like the other data messages, a data keyframe contains a handler name and a list of properties that store the data. There is no limit to the number of data keyframes that can be set. For security reasons, only the publisher and the server are allowed to set and clear the data keyframes.

Data keyframes are sent from the client through `NetStream.send()` or `Stream.send()` in the server-side script. Two special methods, `@setDataFrame` and `@clearDataFrame` are defined to set and clear the data keyframe respectively. To avoid collision with other client defined methods, an `@` sign is added to these methods.

Multipoint publish

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later.

Another challenge for live broadcasting is scalability. Due to the limitations of processor resources or network bandwidth, one server can only support a fixed number of subscribers. To support more simultaneous viewers, some of the traffic needs to be handled by other servers. Multipoint publish allows the broadcaster to publish to multiple servers with only one client-to-server connection.

In the past, the API in the server-side script would only allow streaming from a remote server in one direction through the use of server `Stream` and `NetConnection` objects. This made it possible to play a stream from a remote server. However, there was no way to remotely publish a stream to a remote server or to know which stream(s) were attempting to be published to another server, unless one-to-one connections were maintained between all servers. In addition, due to restricted server-side script access on content delivery networks, stream republishing was not possible.

With the new APIs available in Flash Media Interactive Server 3, broadcasters can dynamically redirect streaming data to another server when necessary, without maintaining a persistent connection between servers and without custom server-side code. New server-side notification, `Publish` and `Unpublish` events, are now available for those who require additional customization.

Multipoint publishing provides a much-needed degree of flexibility, customization, and scalability to large-scale live video applications. For more details, consult the *Flash Media Server Developer Guide*.

Securing content with Flash Media Server 3

Whenever content is distributed electronically, there is some risk of it being copied, misappropriated, or redistributed. Flash Media Server offers several levels of security to protect your content and server resources that are unobtrusive, intuitive, and convenient to consumers.

Content vulnerabilities

There are a number of ways that online digital content can be compromised:

- **Raiding the browser cache**—Though the filenames are not easily read, it is relatively simple to retrieve video files from the browser cache. (This vulnerability is only present with progressive video delivery; streams are never cached.)
- **Video URI access**—Video URIs can easily be discovered using free "sniffer" utilities.
- **SWF re-serving**—Your SWF can be copied and re-served from another domain. SWFs can also be decompiled, often revealing your Flash Media Server address, application and stream names.
- **Replay technologies**—Also referred to as "stream ripping," this is the most insidious of security issues because it is more difficult to prevent. Stream ripping utilities actually intercept the data stream and record it to a file that can then be played.

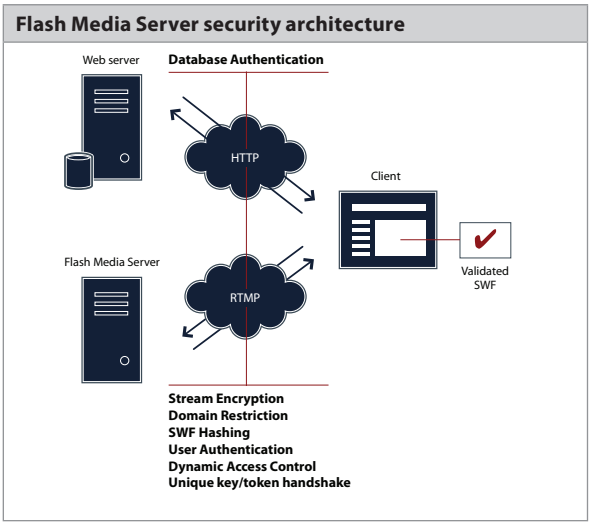
Flash Media Server security architecture

As discussed earlier, streaming has a higher level of security than progressive delivery, since media files are never cached to disk. Flash Media Server further enhances protection against other risks with a number of additional security features:

- User authentication using server-side ActionScript
- Authorization adaptor
- Access adaptor
- SWF verification
- Domain access control
- Custom solutions offered by content delivery networks
- Stream encryption using RTMPE or RTMPS

First, we'll look at the overall Flash Media Server security architecture (shown in the following figure) and then examine each of the protection measures in depth.

Flash Media Server security architecture



Locking down your content

Regardless of the sensitivity or ownership of your content, you'll want to implement some level of security when deploying to the web. It's best to begin by securing your server, then securing your content. Let's examine each of the security measures you can take in more detail.

Restrict access from domains

By default, a client can connect to Flash Media Server from any domain or IP address, which can be a security risk. You can create a whitelist of allowed domains (or a blacklist of banned domains) to ensure that only authorized clients can connect to your applications or services. You can add a comma-delimited list of domains and/or IP-address blocks in the `Adaptor.xml` or `vHost.xml` configuration files to add this level of security. This is usually the first step in locking down your server; it prevents malicious or unauthorized domains from freely accessing your applications and streams.

User authentication

There are several methods of user authentication available with Flash Media Server 3. We will discuss the server-side ActionScript method.

Server-side ActionScript

The next step to increase security would be to implement a user authentication scheme to validate the connecting client. For example, using variables passed in through the client NetConnection method, you could implement a simple username/password, an encrypted token (MD5 Hash), or a unique key:

- User credentials (login/password):

```
NetConnection.connect("rtmp...", "username", "password");
```

- Encrypted token (MD5 Hash):

```
NetConnection.connect("rtmp...", 6aef79f07bc8f23c38e8979f3630f436);
```

- Unique key:

```
NetConnection.connect("rtmp...", 349jh3k4324h9.234234098);
```

Then, on the server-side, Flash Media Server would be able to integrate with web services (SOAP), Flash Remoting, XML, HTTP Post (`loadVars`) or simple file access, to validate the client based on the data sent. This authentication scheme could be as simple as checking login information against a database, or as complex as creating an SSL-based token system using ColdFusion.

Access adaptor plug-in

Improved feature in Flash Media Interactive Server; requires Flash Player 6 or later.

An access adaptor is a server plug-in written in C++ that intercepts connections to the server, and determines whether requests should be accepted, rejected, or redirected before the requests reach the server's script layer. You can create custom logic in the access adaptor to handle client connection requests. For example, you could query your account database upon client login, and then update the database record after the client connection was accepted.

The access adaptor can be configured to accept or reject requests based on the number of clients currently connected or the amount of bandwidth currently being consumed. You can also set read and write access for files and folders on the server, set permissions to access audio and video bitmap data, and inspect client properties through the access adaptor.

When you use the access adaptor, you are actually catching the connection before it is processed by Flash Media Server. For this reason, you are limited to trapping only the connection events. If you want to apply additional rules after the connection is established, you would need to configure an authorization adaptor.

Note: There can only be one Access plug-in per Flash Media Interactive Server installation.

Authorization adaptor plug-in

New feature in Flash Media Interactive Server; requires Flash Player 6 or later.

The next line of defense is the authorization adaptor. A server plug-in written in C++, the authorization adaptor authorizes client access to server events. Once the connection has been established, but before it is accepted, the authorization adaptor comes into play.

Authorization adaptors can:

- Authorize connections to the server
- Authorize playing a stream or seeking in a stream
- Authorize publishing a stream
- Disconnect clients from the server
- Call a method in server-side ActionScript
- Deliver content to clients according to their geographic location, subscription level, and stream origin
- Limit time and duration of a user's access to specific streams
- Map a logical stream path to a physical stream path. For example, a client requests the stream "foo.flv," but since he is not a premium member of the service, he should only receive the low-quality version of that content, so he is actually served "bar.flv."

Unlike the access adaptor, you can use multiple authorization adaptors to sequentially perform actions on the incoming event. For example, `auth1.dll` (or `auth1.so`) could authorize the client connection; `auth2.dll` (or `auth2.so`) could then authorize that client to publish a stream, and so on. The server applies the adaptors in alphabetical order.

As you can see, authorization adaptors can be very powerful for stream security and access control at a granular level. They can be configured to implement custom functionality ranging from rights management to logging.

Dynamic access control

When clients access the server, they have full access to all streams and shared objects by default. Access control is possible, however, using server-side ActionScript. You can create a dynamic access control list (ACL) which controls who has access to read, create, or update shared objects or streams.

In server-side ActionScript, each client that connects is assigned to a Client object. Each Client object has `readAccess` and `writeAccess` properties. These properties can accept multiple comma-delimited values. By setting these values when you accept the client connection, you can control which streams and shared objects any given client can access.

Stream encryption

Flash Media Server 3 offers two options for encrypting your streams: SSL and RTMPE.

SSL

Feature available in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 8 or later.

In earlier versions of Flash Media Server, encrypted streaming was available using SSL delivery, through RTMPS. This form of encryption is still supported in Flash Media Server 3. Implementation requires the use of a third-party certificate with some server-side configuration. Flash Media Server 3 now offers an easier, optimized way to implement an encryption solution, using encrypted RTMP (RTMPE).

RTMPE

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 9,0,115,0 or later.

Encrypted RTMP (RTMPE) is enabled on Flash Media Server by default, and allows you to send streams over an encrypted connection without requiring certificate management. Offering secure 128-bit encryption, RTMPE is only supported in Flash Player 9 or later, with the updated FLVPlayback component and `NetConnection` classes. Both SSL and RTMPE can also be "tunneled" to ensure connectivity through network firewalls. RTMPE is the recommended form of encryption, as it is easier to deploy and is much faster than SSL.

Implementing stream encryption in your applications is easy. Simply specify the protocol when you connect to your application:

- SSL

```
NetConnection.connect("rtmps://yourFMSserver.com");
```

- Tunneled SSL

```
NetConnection.connect("rtmpts://yourFMSserver.com");
```

- Enhanced RTMP

```
NetConnection.connect("rtmpe://yourFMSserver.com");
```

- Tunneled enhanced RTMP

```
NetConnection.connect("rtmpte://yourFMSserver.com");
```

Defend against replay technologies

Replay technologies or "stream ripping" has been a difficult security issue to solve because it allows the viewer to directly access and record the data of a stream.

Stream encryption prevents stream ripping. In the past, SSL was the only choice, and it was too slow for most applications. Flash Media Server 3 uses RTMPE, which is much more efficient and easier to implement.

Another method of defense against stream ripping is to insert intelligence into your server-client communications. By adding additional code to your video player, you could require your SWF to respond to a request from Flash Media Server to verify a unique string sent from the server, for example. This interrupts the flow of data to the stream ripping software, as it cannot respond with the correct data, and will be denied access.

Digital Rights Management support

New feature in Flash Media Streaming Server and Flash Media Interactive Server; requires Flash Player 6 or later. RTMPE/SWF verification requires Flash Player 9,0,115,0 or later.

DRM has two key elements: encryption and access control. There are two ways to deliver video to a consumer: streaming or downloading. When you stream video from Flash Media Server, you immediately increase your protection.

Encryption with Flash Media Server is done in real-time with RTMPS (SSL) or with RTMPE in Flash Media Server 3.

Access control with Flash Media Server is done simply with SWF verification. Access control is much more powerful with Flash Media Interactive Server because of its new plug-in architecture, along with the server-side application layer. Using web services (SOAP), Flash Remoting, or XML you can create a system with secure tokens that provide access control over your content.

These are the basic principles of DRM for streaming. For the download use case, Adobe will be releasing new technology with the Adobe Media Player in early 2008.

Content protection from CDNs

An easy way to add content protection to your streaming content is to use FVSS through Adobe's CDN partners. Many of Adobe's FVSS partners offer plug-and-play restricted access and secure video streaming solutions.

To learn more about how a CDN can help protect your content, visit the Adobe FVSS website: *www.adobe.com/go/fvss*.

Glossary

Adobe AIR: A cross-platform tool that enables developers to use their existing web development skills in HTML, AJAX, Flash, and Flex to build and deploy rich Internet applications to the desktop.

Adobe Media Player: A desktop media player that brings the best of both the broadcast television and web video worlds to your desktop—providing high-quality content both online and offline, with a wide range of business model possibilities.

Bandwidth: Refers to the amount of throughput of a server or a client computer. Usually measured in megabits per second (Mbps) or kilobits per second (Kbps). A typical wired Ethernet connection is 100Mbps and WiFi is 54Mbps. Server and Client bandwidth limits determine how much video can be served or received.

Buffer: The amount of video stored in RAM on the client computer. The larger the buffer, the smoother the video will play back. The buffer is never written to disk.

Content Delivery Network (CDN): These companies offer streaming services and bandwidth so customers do not need to set up and install servers of their own.

Client: The consumer connecting to Flash Media Server via Flash Player or Adobe AIR applications.

Codec: The format in which a video or audio file is encoded. Flash uses Sorenson Spark, On2 VP6-S, On2 VP6-E, and H.264 codecs for video; Nellymoser, MP3, and AAC for audio. Short for "code/decode," the decoding part of the codec must be present in the player to play back video using a specific codec.

Connection: When a client is streaming video it consumes one connection. When multiple clients are streaming at the same time, they are referred to as simultaneous connections.

Content: Video or audio data streamed from Flash Media Server.

Digital Rights Management (DRM): Video encoded with DRM can be sold and protected against stealing and unauthorized sharing.

Encoder: Software that compresses or transcodes video from one format to another.

Enhanced-RTMP: The next-generation Real Time Messaging Protocol (RTMP) that increases security and performance.

Flash Lite 3: The next-generation mobile Flash player that will support the VP6/Spark codec and allow for RTMP connections to Flash Media Server.

Flex: Adobe Flex is a cross platform, open source framework for creating rich Internet applications that run identically in all major browsers and operating systems.

Flash Media Encoder: A free Windows XP-based desktop application that connects to Flash Media Server and allows you to stream live video and audio to Flash Player.

Flash Media Solution Provider program: A partner program that helps promote a strong ecosystem around Flash Video and Flash Media Server.

Flash Video Streaming Service (FVSS): Adobe has partnered with leading CDN providers to offer hosted services for delivering on-demand video for Flash Player across high-performance, reliable networks.

Live: Live Flash streaming using Flash Media Encoder or Flash Player.

Publishing Point: A directory on Flash Media Server where customers can place video/audio content and publish live video.

On2 VP6: A video codec that offers high quality, lightweight full-screen playback (available since Flash Player 8). VP6-S is a simplified version that is ideal for delivery of high-quality video to older computers (available in Flash Player 9 or later). VP6-E, the original codec that shipped with Flash Player 8, offers slightly higher quality and requires more processing power for playback.

Real Time Message Protocol (RTMP): Adobe's proprietary method of communication between Flash Player clients and Flash Media Server.

Quality of Service (QOS): Refers to the quality of the consumer's playback experience.

Solution Provider: Consulting/enablement organizations that provide advanced knowledge of Flash Media Server and video for Adobe Flash Player, and their integration over multiple devices.

Sorenson Spark: Original video codec in Flash Player 6 and 7. An encoder for this codec is also built into Flash Player, allowing for webcam broadcast and archiving when used with Flash Media Server.

Transcoding: The conversion from one video format to another. Usually transcoding allows you change the codec. Each time a file is transcoded, quality is lost.

Video on Demand (VOD): A term used to describe the delivery of prerecorded Flash video streaming.

Online resources

Flash On™ (Adobe video showcase)
www.adobe.com/flashon

HD Video with Adobe Flash
www.adobe.com/products/hdvideo

Adobe HD Gallery
www.adobe.com/products/hdvideo/hdgallery

Understanding the difference between progressive download and streaming video
www.adobe.com/devnet/flash/articles/flv_download.html
www.adobe.com/products/hdvideo/supported_technologies/streaming.html

DRM and digital media protection with Flash Media Server
www.adobe.com/devnet/flashmediaserver/articles/digital_media_protection.html
www.adobe.com/products/hdvideo/supported_technologies/content_protection.html

Performance-tuning Flash Media Server 2 for live webcasts using Linux
www.adobe.com/devnet/flashmediaserver/articles/performance_tuning_webcasts.html

Exploring Flash Player support for high-definition H.264 video and AAC audio
www.adobe.com/devnet/flashplayer/articles/hd_video_flash_player.html

AAC-HE version 2 details
www.adobe.com/products/hdvideo/supported_technologies/heaacv2.html

H.264 details
www.adobe.com/products/hdvideo/supported_technologies/h264.html

Streaming Through Firewalls and Proxies: From the Client's Perspective
www.adobe.com/devnet/flashcom/articles/firewall_streaming.html

Flash Media Encoder 2
www.adobe.com/products/flashmediaserver/flashmediaencoder

Adobe FVSS partners
www.adobe.com/go/fvss

Flash Media Solution Provider program
www.adobe.com/go/fmsp

Flash Player 9 Update FAQ
http://labs.adobe.com/wiki/index.php/Flash_Player:9:Update:H.264

Flash Media Server community

FMSGuru.com
www.fmsguru.com

FlashComGuru
www.flashcomguru.com

FlashConnections
www.flashconnections.com

Flash Video | Optimizations and Tools
http://flashvideo.progettosinergia.com/

References

A Streaming Media Primer
www.adobe.com/products/aftereffects/pdfs/AdobeStr.pdf

A Digital Video Primer: Understanding and Using High-Definition Video
www.adobe.com/designcenter/productionstudio/articles/prs1ip_hdprimer/prs1ip_hdprimer.pdf

Video content protection measures enabled by Flash Media Server
www.adobe.com/devnet/flashmediaserver/articles/protecting_video_fms.html

About the author

Lisa Larson-Kelley is a developer, consultant, and teacher on subjects related to electronic media. She coauthored the book F*lash Video for Professionals* (Wiley, 2007). Her blog is at *www.flashconnections.com*.

Adobe

Adobe Systems Incorporated
345 Park Avenue
San Jose, CA 95110-2704
USA
www.adobe.com

Adobe, the Adobe logo, ActionScript, Adobe AIR, ColdFusion, Flash, Flash Lite, Flex, Flex Builder, Macromedia, and "Flash On" are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Mac is a trademark of Apple Inc., registered in the U.S. and other countries. HP is a registered trademark of Hewlett-Packard Company. Intel, Intel Xeon, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries. Linux is the registered trademark of Linus Torvalds in the U.S. and other countries. Microsoft, Windows, Windows Server, and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Red Hat is a trademark or registered trademark of Red Hat, Inc. in the United States and other countries. All other trademarks are the property of their respective owners.

© 2008 Adobe Systems Incorporated. All rights reserved. Printed in the USA.

95010395 2/08